

H2kInfosys

H2K Infosys provides online IT training and placement services worldwide.

www.H2KINFOSYS.com

USA- +1-(770)-777-1269, UK – (020) 3371 7615

Training@H2KINFOSYS.com / H2KInfosys@gmail.com

DISCLAIMER

H2K Infosys, LLC (hereinafter “H2K”) acknowledges the proprietary rights of the trademarks and products names of other companies mentioned in any of the training material including but not limited to the handouts, written material, videos, power point presentations, etc. All such training materials are provided to H2K students for learning purposes only. H2K students shall not use such materials for their private gain nor can they sell any such materials to a third party. Some of the examples provided in any such training materials may not be owned by H2K and as such H2K does not claim any proprietary rights for the same. H2K does not guarantee nor is it responsible for such products and projects. H2K acknowledges that any such information or product that has been lawfully received from third party source is free from restriction and without any breach or violation of law whatsoever.

QUALITY ASSURANCE (QA) INTERVIEW QUESTIONS AND ANSWERS

1. Can you tell me about yourself?

Answer: In my QA career, I have been working on various system platforms and operating systems like Windows 95, Windows 2000, Windows XP and UNIX. I have tested applications developed in Java, C++, Visual Basic and so on. I have tested Web-based applications as well as client server applications.

As a QA person, I have written Test Plans, Test Cases, attended walkthrough meetings with the Business Analysts, Project Managers, Business Managers and QA Leads. Attended requirement review meetings and provided feedback to the Business Analysts. I have worked in different databases like Oracle and DB2, wrote SQL queries to retrieve data from the database.

As far as different types of testing is concerned, I have performed Smoke Testing, Functional Testing, Backend Testing, BlackBox Testing, Integration Testing, Regression Testing and UAT (User Acceptance Testing) Testing. I have participated in Load Testing and Stress Testing.

I have written defects as they are found using ClearQuest and TestDirector and now QC. Once the defects were fixed, retested them and if the passed, closed them. If the defects were not fixed, then reopened them. I have also attended the defect assessment meetings as necessary.

In the meantime, a continuous interaction with developers was necessary.

This is pretty much what I have been doing as a QA person.

2. What did you do in your last project?

In my last project, the application was a web-based application developed in Java platform. As a QA Person, I wrote Test Plans from the requirement documents and Use Cases. I performed Smoke Testing, Functional Testing, Backend Testing, BlackBox Testing, Integration Testing, Regression Testing and UAT (User Acceptance Testing). I have participated in Load Testing and Stress Testing. I

attended several walkthrough meetings for requirement reviews and provided feedback to the Business Analysts. Mostly, I was in the backend testing, which required writing SQL queries directly to the database.

Besides these, I wrote defects using ClearQuest. Once the defects were fixed, retested them and if they passed, closed them. If the defects were not fixed, then reopened them.

3. Have you written Test Plan? What is a Test Plan? What does it include?

Yes.

What is a Test Plan?

A Test Plan is a document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks and who will do each task (roles and responsibilities) and any risks and its solutions.

What does it include? A Test Plan includes Heading, Revision History, Table of Contents, Introduction, Scope, Approach, Overview, different types of testing that will be carried out, what software and hardware will be required, issues, risks, assumptions and sign off section.

Click here to see how a complete **Test Plan Sample** looks like.

4. Have you written a Test Case?

Yes.

What is a Test Case? What does it include?

A Test Case is a document that describes step by step process how to test the application. A Test Case includes Test Case ID, Steps Description, Expected Output, Actual Output, Pass/Fail, Remarks.

5. How many Test Cases did you write in your last project?

Answer: I wrote about 1100 Test Cases in my last project. (The reasonable number of Test Cases varies from 500 to thousands. The

number 1100 test cases can be completed in a 6 month project duration).

6. What document did you refer to write the Test Cases?

Requirement document. (NOTE: It can also be Use Cases, or Design Document)

(Note: It depends company to company. In some companies, they use Use Cases. In some companies, they use Requirement Documents and in some companies, they use Design Document. However, in practical scenario, most of the companies have requirement document at least). This is the sample Requirement Document for Mercury Tours.

7. Did you have a situation where you did not have any documents (no requirement document, no Use Cases, or no Design Document) and you had to write the Test Cases? How did you write the Test Cases?

Yes. I have been to that kind of scenarios several times. There were companies where they had no documents at all. In that case, I had to discuss the application scenario and functionalities with the Business Analysts or developer. I kind of prepared a document in consultation with Business Analysts and Developers and then started writing Test Cases.

8. Have you worked with the Uses Cases before?

Yes. I have written Test Cases using Use Cases.

Can you tell me what a Use Case is?

A use case is a document that describes the user action and system response for a particular functionality. (you can also include, For example, in the Use Case given below, is a Use Case for login system for a company called Auto Parts One. This application is being developed by Digital Systems, Inc. The project name is Auto Parts One. However, the business owner (user) is a company called American Auto Parts of the North (imaginary name). Or

What is a Use Case and what does it include?

A Use Case is a document that describes the user action and system response for a particular functionality. It includes cover page, Revision History, Table of Contents, Flow of Events (normal flow and alternative flow), Exceptions, Special Requirements, Pre-conditions and Post-conditions.

Now, Let us write Test Cases based on this Use Case. Remember, one Use Case can have many Test Cases. For example, look below:

For a complete Test Case for www.digitalsystemsllc.com, please click here.

9. What is Software Development Life Cycle?

The systems (or software) development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application.

It includes the following different stages:

- 1. Requirement phase**
- 2. Design phase**
- 3. Coding (programming)**
- 4. Testing**
- 5. Release (Production)**
- 6. Maintenance (Support)**

10. What is Business Requirement Document (BRD)?

It is a document that describes the details of the application functionalities which is required by the user. This document is written by the Business Analysts.

What is Software Testing Life Cycle (STLC)?

The testing of software has its own life cycle. It starts with study and analyzing the requirements. Here is the software testing life cycle:

1. Requirement Study
2. Test Planning
3. Writing Test Cases
4. Review the Test Cases
5. Executing the Test Cases
6. Bug logging and tracking
7. Close or Reopen bugs

What is Business Design Document?

It is the document which describes the application functionalities of the user in detail. This document is the further details of the **Business Requirement Document**. This is a very crucial step in the SDLC. Sometimes the Business Requirement Document and Business Design Document can be lumped together to make only one Business Requirement Document.

What is Code Generation or Program?

Coding is the process of translating the Business Design Document into the machine readable form. If the design is done in detailed manner, the Code Generation can be done without much application. Programming tools like Compilers, Interpreters and Debuggers are used to generate the code thru different high level language like C, C++, Pascal, Java.

11. What is a Module?

A 'Module' is a software component that has a specific task. It can be a 'link' which can go inside to its component detail.

12. What is meant by Walk-thru meeting?

Before start working in a module and/or after accomplishing the testing of a module, the tester calls a meeting to disseminate his findings or to share his queries to other tester or leads of the company working on the same application that is called the Walk-thru meeting.

13. What is Build?

When each of the different modules of software is prepared, they are put in a single folder by the Configuration Management Team (CMT) and it is called the 'Build'. In other word, the developers put their code in the shared location (folder) and all those code (modules) are combined together so that it is a complete application that works.

What is meant by the Build Deployment?

When the Build so prepared by the CMT is sent to different Test Environments, it is called the Build Deployment.

14. What is Test Strategy?

A test strategy is an outline that describes the testing portion of the software development cycle. It is created to inform project managers, testers, and developers about some key issues of the testing process. This includes the testing objective, methods of testing new functions, total time and resources required for the project, and the testing environment.

The test strategy describes how the product risks of the stakeholders are mitigated at the test-level, which types of test are to be performed, and which entry and exit criteria apply. (source: Wikipedia)

The test strategy is created based on development design documents.. It is written by the Test Manager or Lead.

The following are some of the components that the Test Strategy includes:

1 Test Levels. 2 Roles and Responsibilities. 3 Environment Requirements. 4 Testing Tools.5 Risks and Mitigation. 6 Test Schedule. 7 Regression Test Approach. 8 Test Groups. 9 Test Priorities. 10 Test Status Collections and Reporting. 11 Test Records Maintenance. 12 Requirements traceability matrix. 13 Test Summary

Are Test Plan and Test Strategy same type of document?

No. They are different documents. Test Plan is a document that collects and organizes test cases by functional areas and/or types of testing in a form that can be presented to the other teams and/or customer where as the Test Strategy is the documented approach to testing. Test Plan is prepared by the tester whereas the Test Strategy is prepared by the QA Manager or QA lead.

Both are important pieces of Quality Assurance processes since they help communicate the test approach scope and ensure test coverage while improving the efficiency of the testing effort.

15. What does the Test Strategy include?

It includes introduction, scope, resource and schedule for test activities, acceptance criteria, test environment, test tools, test priorities, test planning, executing a test pass and types of test to be performed.

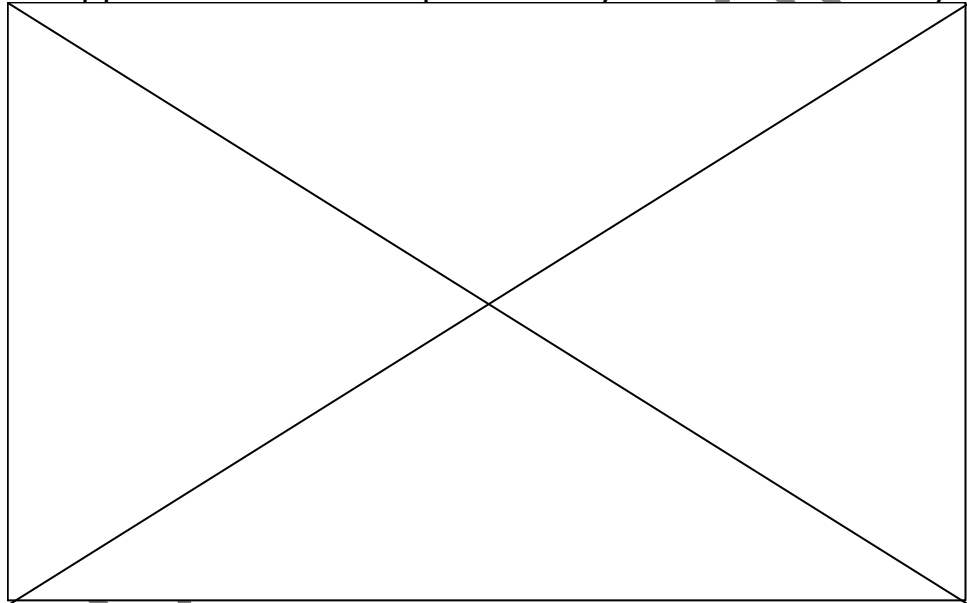
16. What are different types of software testing?

Different types of testing carried out are:

- 1) **Unit testing**
- 2) **Shakeout testing**
- 3) **Smoke testing (Ad-hoc testing)**
- 4) **Functional testing**
- 5) **Integration testing**
- 6) **Regression testing**
- 7) **System testing**
- 8) **Load testing**
- 9) **Stress testing**
- 10) **Performance testing**
- 11) **User acceptance testing**
- 12) **Black box testing**
- 13) **White box testing**
- 14) **Alpha testing**
- 15) **Beta testing**

Note: Except the Shakeout testing and Unit testing which are respectively done by the CMT and Coder/Developer, all other testing are done by the QA Engineer (Tester).

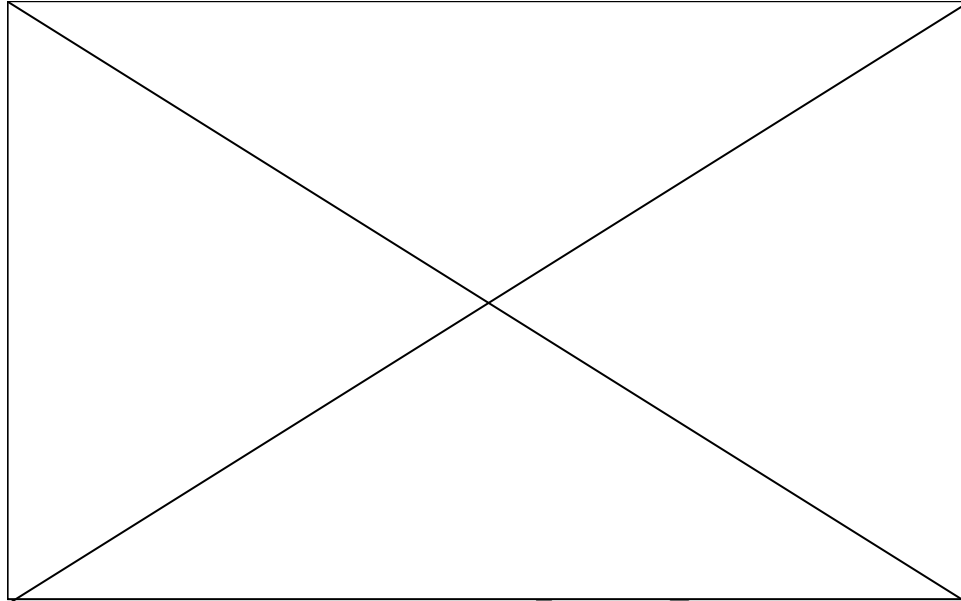
- 1) **Unit testing**: It is a test to check the code whether it is properly working or not as per the requirement. It is done by the developers (Not testers).
- 2) **Shakeout testing**: This test is basically carried out to check the networking facility, database connectivity and the integration of modules. (It is done by the Configuration Team)
- 3) **Smoke testing**: It is an initial set of test to check whether the major functionalities are working or not and also to check the major breakdowns in the application. It is the preliminary test carried out by



the SQA tester.

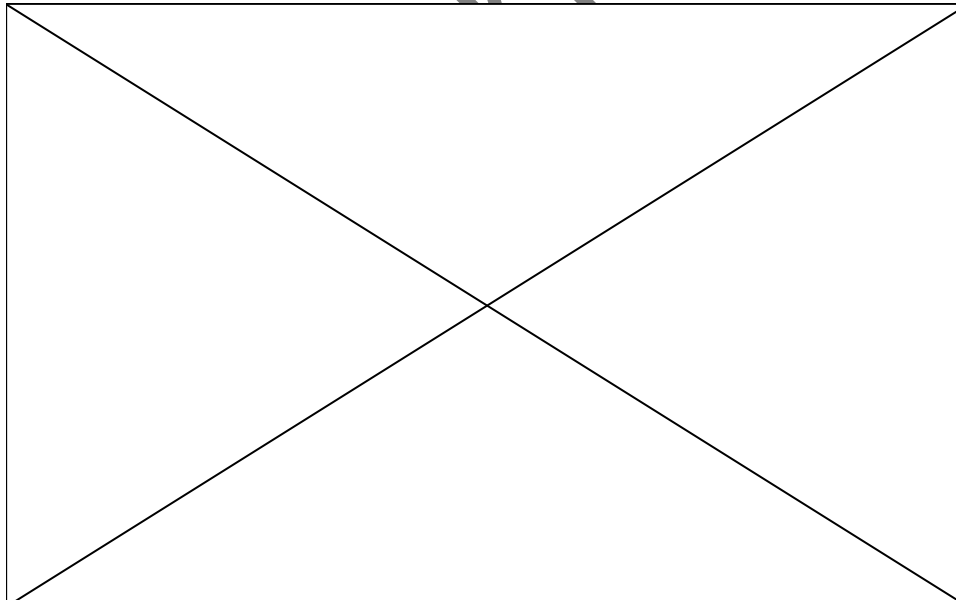
- 4) **Functional testing**: It is a test to check whether each and every functionality of that application is working as per the requirement. It is major test where 80% of the tests are done. In this test, the Test Cases are 'executed'.
- 5) **Integration testing**: It is a test to check whether all the modules are combined together or not and working successfully as specified in

the requirement



document.

6) **Regression testing:** When functionality is added to an application, we need to make sure that the newly added functionality does not break the application. In order to make it sure, we perform a repeated testing which is called Regression Testing. We also do regression testing after the developers fix the bugs. See the video below for more understanding. (Courtesy of guru99.com).



- 7) **System testing:** Testing which is based on overall requirements specification and it covers all combined parts of a system. It is also a black box type of testing.
- 8) **Load testing:** It is a test to check the user's response time of number of users using any one scenario (single business process) of the same application at the same time.
- 9) **Stress testing:** In this type of testing the application is tested against heavy load such as complex numerical values, large number of inputs, large number of queries etc. which checks for the stress/load the applications can withstand.
- 10) **Performance testing:** It is a test to check the user's response time of number of users using multiple scenarios (multiple business process) of the same application at the same time.
- 11) **User acceptance testing:** In this type of testing, the software is handed over to the user in order to find out if the software meets the user expectations and works as it is expected to.
- 12) **Black box testing:** It is test where a tester performs testing without looking into the code. OR A testing method where the application under test is viewed as a black box and the internal behavior of the program is completely ignored. Testing occurs based upon the external specifications. Also known as behavioral testing, since only the external behavior of the program is evaluated and analyzed.
- 13) **White box testing:** It is a test where a tester looks into the code and performs the testing.
- 14) **Alpha testing:** In this type of testing, the users are invited at the development center where they use the application and the developers note every particular input or action carried out by the user. Any type of abnormal behavior of the system is noted and rectified by the developers.
- 15) **Beta testing:** In this type of testing, the software is distributed as a beta version to the users and users test the application at their sites. As the users explore the software, in case if any exception/defect occurs that is reported to the developers.

What is Negative Testing?

Testing the system or application using negative data is called negative testing, for example, testing password entering 6 characters where it should be 8 characters should display a message.

When we test an application by putting negative values (instead of actual values), then the system should not allow the other values rather than the actual value. The system should give an message that the value is not correct. This is called negative testing.

Another example is, if a user tries to type a letter in a numeric field, the correct behavior in this case would be to display the "Incorrect data type, please enter a number" message. The purpose of negative testing is to detect such situations and prevent applications from crashing. Also, negative testing helps you improve the quality of your application and find its weak points. (source: Jerry Ruban)

What is the difference between Load Testing and Performance Testing?

Basically Load, Stress and Performance Testing are the same. However, Load testing is the test to check the users' response time of number of users of any one scenario of the application whereas Performance Testing is the test to check the user response time for multiple scenario of the same application.

17. What was the process of QA testing in your company where you worked for the last time? (or As far as the QA process is involved, what was the testing process in your company?)

The QA testing process that was followed in my last company where I worked was like this: First of all the Business Requirement Document was prepared as per the client's requirement (with the muck-up screen shots). Then on the basis of the requirement document, Test Strategy, Test Plans and Test Cases were written in sequential order. Once the Build is made and deployed to the different testing environments where different types of testing were performed to check whether there are any defects.

18. What is SQL?

SQL stands for Structured Query Language. SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and update data in a database. SQL works with

database programs like MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc.

Unfortunately, there are many different versions of the SQL language, but to be in compliance with the ANSI standard, they must support the same major keywords in a similar manner (such as SELECT, UPDATE, DELETE, INSERT, WHERE, and others).

Note: Most of the SQL database programs also have their own proprietary extensions in addition to the SQL standard.

Where do you write SQL query?

We write SQL queries using some these tools: Todd, Squirrel and Rapid SQL.

Do you really need to write SQL as a QA Engineer?

Yes. You need to. No matter whether it is a small company or big, they have a database and you need to validate the data by writing SQL queries going into the database. The stronger you are in SQL, the better the chance of getting a job.

What are the basic commands in SQL+?

They are:

SQL>select *from tab;	-to directory of database
tables	
SQL>ed	-to edit the queries in
the notepad	
SQL>/	-to run or execute the
query command	
SQL>create table 'table name'	-to create a table
SQL>desc 'table name'	-to display table with column
name with type	
SQL>alter table 'table name'	-to add a column
name' 'type'	
SQL>alter table 'table name'	-to modify the name and type of
a columnmodify 'column name' 'type'	

What is the most common syntax you have used while writing SQL query?

Answer: SELECT

What is a Primary Key?

In a database table, the Primary Key is a column which has a unique value for each of the row within that column. It can't have NULL value.

What is a Unique Key?

In a database table, the Unique Key is a column which may or may not have null value of each of the row within that column.

What is Data?

Data is number, character or image which has some information.

What is Database?

It is collection of logically related data designed in a tabular form to meet the information needs of one or more users.

19. What is Change Control (OR Change Request)?

Answer: It is a document that describes the additional functionalities that are added after the Business Requirement Document is signed off. It can be updated in the old business requirement document or it can be a separate document.

(For example, in the Business Requirement Document, on the login page, there are User Name and Password fields. The owner of the software wants to add, "If you do not have User Name and Password, please click here." This is a change. But this change came after the document is signed off by the Project Managers. Now this is a change control and comes as a separate document. (It is also called Change Request, Modification Request).

20. Have you written Change Control?

Answer: Yes. There was a situation where in one page of an application in my previous project, when the user clicked "Contact" link, it would pop up a different window (new separate window). But it was NOT the way it was described in the requirement document. In

the requirement document, when the user clicks “Contact” link, then it should navigate to another page (Not a separate new window. Then was it a problem? Functionality wise, it was NOT a problem, however, on all the other pages, when the user clicked “Contact” link, the system would navigate to next page (not a separate window). So, it was NOT CONSISTENT with the other functionalities on the other pages. Therefore, it was a consistency issue. I reported this as a bug. But the Project Manager asked me to write it as a Change Control (because it requires more budget to fix this issue) so that he can address this issue at a later time. So I wrote this as a Change Control. (However, it is NOT a job of a tester to write change control. It’s the business analyst’s job)

20. What is Backend Testing?

It is a test to check whether the data displayed in the GUI front end report format matches with the particular data in the original database.

21. Have you done any Back End Testing and/or if you did, how did you do it in your last project?

Yes I did. I was working on Reports. When I was working in my last project, this was my scenario:

It was the case of testing one part of application used in the bank, where a customer comes to a bank’s front desk associate and ask for opening an account. The associate then asks for the personal information about the customer which, are the primary data, such as: First Name, Last Name, Date of Birth, Address and Social Security Number. The associate then put these primary data of that particular customer into the computer, which then afterwards batch-processed into the DATABASE in XML Format. Then the batch-processed data is sent to ETL (Extract-Transform-Load, which is software made by ‘AbInitio’ or ‘Informatica’) which processes the job to create a file to produce the report. The file is displayed to a GUI Front End report format with the help of Crystal Report/Business Object. In the GUI Front End report, let us say, if for January, the income of that person was displayed as \$ 900.00, then my job was to validate this data by writing SQL queries whether this displayed data matches with the original input data in the database, being called as the Back End Testing.

How can you be sure that the query you wrote is correct? Or how do you know that the data you pulled from the database is correct?

Answer: I write SQL query based on the requirement document. In the requirement document, various conditions are given for the query. Based on those conditions, I write SQL query. Therefore, anything different from the requirement document is definitely a defect.

22. What is XML?

- XML stands for EXtensible Markup Language.
- XML is a markup language much like HTML.
- XML was designed to describe data.
- XML tags are not predefined and we must define our own tags.
- XML uses a Document Type Definition (DTD) or an XML Schema to describe data.
- XML with a DTD or XML Schema is designed to be self-descriptive.
- XML is a W3C Recommendation.

23. From your resume, I see that you have been working in one place for a very short period of time. This raises me questions why. Can you explain why?

Ans. As a consultant, I am hired for a certain period of time, normally for 6 months to 1 year. Once the project is over, I needed to move to another project. That's why you see me in the resume jumping frequently here and there.

24 What do you do on your first day of the work?

(Note: The person who is asking this question probably wants to know how the real scenario of a working person at work. It is a hard question for those who has never worked in a work place as a Software Tester.)

Answer: On the first day, normally, we will be given a computer and support people will set up the User Name and Password for the computer. If that is done already, then the QA Lead or QA Manager will give me a brief walk through of the documents (which documents are where), introduce to different team members (normally to the

ones you will be working with). Then your boss will ask you to step into work what needs to be done. However, the first thing normally is, they will ask you to read the documents available for that project.

What do you do if you have any questions to ask? Who do you ask?

At the beginning, we all panic, what kind of questions to ask? What if they ask questions that I don't know? Is it OK to ask questions? What do I do if I don't know how to do the job I am assigned to? and so on. As mentioned earlier, on the first day, your Manager will give you the system (computer) (They normally call system, not computer), will tell you what the User ID and Password is, where are the QA documents on the shared drive (or Network drive) are and so on. They will definitely ask you to read a lot of documents at the beginning (And you must read read and read those documents AS MUCH AS POSSIBLE. At the beginning, allocate about 2 hours extra at home for reading these documents. This habit will put you on the top of your job). These documents are normally design specification document (DSD). Different companies call it with different names, for example, Requirement Specification Document (RSD) and so on. After reading the documents, you will be asked to write Test Plans or Test Cases (Don't panic. The Test Plans and Test Cases templates will be give by your manager or test lead and they will tell you what to do and how to do because different companies have different formats they follow. If they don't have one, then you can always prepare a sample from this website (see on the right column) and give it to them. You will be hero)

Who do you ask?

Now let's say you did not understand something while reading documents. Who are you going to ask? Answer-Business Analysts who wrote this document. If you have any other questions that you don't know, you will be asking that to you friend first, if he/she is not able to answer, then ask this question to the Lead (or Manager). Do not ask too many questions (some people get irritated). Therefore, it is important to read read and read. That's the only way to succeed. If you have any questions in QC, or QTP or any other automation tools, then there is a HELP menu as well as tutorial. Please go through these, read them before you ask any questions to anyone else. What kind of questions should I ask in the meeting? Nothing. My advice is, keep your mouth shut. Just listen. This is the

best way to handle the job until you are confident enough to speak and you know what you are talking about. If they ask you some questions, then reply gently, wisely.

How to deal with your team members?

Most probably, you will not be the only tester in the team. There will be more than you. Sometimes, dealing with you team members is frustrating, specially when you are new. They try to ignore you. They want to show themselves smart. Don't worry. Don't blame them. This part of the human nature. Try to cope with it. Invite them when you go for coffee (in the coffee room in your office, don't go outside), try to share your feelings and so on. It is all how you handle your friends. It is part of your daily activities, handle it gently. This is part of the situation I have gone through, my friends have gone through. I am just sharing this with you.

28. Have you used automation tools?

(Normally, when some one asks this question, we tend to think about automation functional testing tools, like WinRunner, LoadRunner, QTP (Quick Test Pro), Rational Robot, Experian and so on. But the reality is, even a Manual Tester also uses automation tools like bug tracking tools like TestDirector, ClearQuest, PVC Tracker and so on. Therefore, your answer should be Yes)

Answer: Yes. I have used TestDirector and ClearQuest as defect tracking tools. (Your answer is based on whether you have used automation tools specially for functional and load testing. If you have NOT used, but read about these tools, then you may be better off saying, "I know about the tools. I was involved in some of the testing using these tools, but would need some brush up in order to work independently." I am saying this because these tools are difficult to tackle in the interview and have to know in depth. In order to pass the interview on functional automation tools, it may not be easy unless you really know the stuff. But, since there is not much to learn in ClearQuest and TestDirector, you only have to know what different types of fields are there in the defect logging window when writing a defect.)

29. When you log a defect using TestDirector (or ClearQuest) what fields do you see?

Answer: When we log a defect, we see Defect ID (it shows later in TestDirector), Summary (where we write short description of the defect), Description (long description of the defect), Detected by (Person who found the defect, it's you), Severity (meaning-is the defect critical? High?Medium? Or Low?), Date, Detected in Version, Priority, Project, Status, Assigned to and so on.

[Click here to see the fields in TestDirector \(go to page 24-27\)](#)

[Click here to see the fields in ClearQuest \(go to page 9\)](#)

30. Are you better working in a team or working alone?

Answer: I am a team player. I get along with team members very well. As far as the working is concerned, I can be equally productive in team or working alone.

(Caution: Never say, I like working alone. This could lead you to not getting a job as they are always looking for people who can get along with other people.)

31. Do you have any situations in the past where you have some arguments with your team members?

Answer: No. I never had that type of situation wherever I have worked.

(Even if you had one, it's a good idea to say "No". This could be a red flag, which might stop you from getting the job)

32. What do you like about a Manager? And what don't you like?

Answer: The best thing I like about a Manager is that the Manager should be able to coordinate with the other teams so that we can get the updated documents, for example, updated requirements documents right away. A Manager who can efficiently in distributes the work to the team, without being biased and easily accessible and protective to his team for the right cause. As far as "what I don't like" is concerned, I don't like a manager who keeps coming to desk 10 times a day to check my work even if it is just a regular work. Once

the responsibility is given, the team member should be trusted and let his work done.

33. Where do you see yourself in another 5 years?

*Answer: I see myself a QA Lead in another 5 years.
(You can also say "QA Manager", but since the QA Manager is taking your interview most of the time, they some times feel challenged.
Therefore, it might be a good idea to limit you to QA Lead)*

34. Why are you in QA?

Answer: I am in QA because I like this job.

35. Why do you like this job?

Answer: I like this job, because it is process oriented. Meaning that I get an opportunity to work from analyzing the requirement documents to writing test plans, test cases, testing the application, logging defects, retesting, preparing reports and finally testing in production as well. Therefore, I am involved from the very beginning to the end of the software development life cycle (SDLC) process. I like this. Another reason is I like to find defects. I enjoy logging defects. The more defects I find, the happier I am.

36. How do you determine what to test in an application?

Answer: First of all we have the test cases (or test scripts) that are written based on the requirement document. This pretty much covers what functionalities to test. Therefore, looking at the test cases tells us what to test in the application.

37. If you have no documentation about the product, how do you test an application? Describe the process.

Answer: Well, this is a situation where I have come across several times. Some of the companies in my previous projects did not have any documents. In this case, I went to the Business Analyst and some times to developers to find out how exactly the functionalities work, how to navigate from one page to another page and so on. After

getting a clear vision, I write test cases based on the conversation (which is a step by step procedure to test an application) and get ready for testing.

What do you do once you find a defect?

Once you find a defect, this is what we need to do:

- 1. **Recreate the Defect:** Once you find a defect, we must try to recreate (meaning that we should be able to reproduce it) at least 3 times so that we are sure that it is a defect. Some times, once we find it log it without recreating, may put us in a false situation (because sometimes the application does not behave in the same way). Therefore, it is important to recreate the same defect several times.*
- 2. **Attach the Screen Shot** (supporting document): Once we confirm that it is a defect, and then it is a good idea to attach supporting documents when we log (write) a defect. For example, screen shot, requirement document etc. For instance, let us say that instead of "Continue" button on a page, there is a typo "Contiinuee". Now, we will make a screen shot of this page (To make screen shot, press "Print Screen" button on the keyboard, and open a Word document, and Click Edit on the Word document and "Past" it. You will see the screen now) Now, a tester needs to write defects in easy and clear language to make all the developers to understand easily.*
- 3. **Log the Defect:** Now, the next step is, we need to log it. Depending on the company what kind of tools they are using (for example, some companies use TestDirector to log defects, some companies use Rational ClearQuest, some use PVC Tracker and so on). If the company is small and cannot afford these expensive tools, then they may simply use Excel sheet to log defects. We log the defect.*

38. What are the basic elements you put in a defect?

Answer: Basic elements we put in a defect are: SEVERITY, PRIORITY, CREATED BY, VERSION NO, HEADER, DESCRIPTION OF THE DEFECT where we write how to recreate a defect, in what module the defect is found, Status, and so on.

39. What is the biggest bug you have ever found?

Answer: Well, there are many big defects I have found in various projects. For example, in the last project, on a page, there was a button called "More Information". Once the user clicked that button, the system would open a new window (pop up).

We could close the new window in 3 ways:

-By clicking X at the top right corner of the page

-By clicking "Close" button on the page

-By pressing combination keys (Alt+F4) on the key board

Although the combination key (Alt+F4) was not mentioned in the test case, I just wanted to try how the application reacts when Alt+F4 is pressed. Then I pressed Alt+F4. The result was a disaster-the application crashed (broke). The application disappeared from the computer monitor. Since it was the last day of testing for us, it brought chaos in our Managers, Leads and the whole teams. Finally, the developers disabled Alt+F4 as a temporary solution and the application went into production.

40. How do you make sure that it is quality software?

Answer: There is a certain process how the quality of software is guaranteed (ensured). It is defined by the 'exit criteria'. (What it means is, a QA Manager writes a document called Test Strategy. This Test Strategy defines the 'exit criteria'.) Exit Criteria gives the measurement, for example, in order to confirm the quality, how many critical defects, high defects, medium defect and low defect are acceptable? These are all defined in the exit criteria. (Normally in practice, for a quality software, there should no critical defects (0 critical), no high defect (0 high), no medium defect (0 medium) and may be 1 low defect)

41. As a QA Tester, can you tell me the situation when you felt the most proud of it?

Answer: When I find the defect that normally others don't find, then I feel very proud. For example, there were situations where I found bugs that crashed the whole system at the end of testing phase. I tried the scenarios where the scenarios were NOT mentioned in the test cases. For example, we can close the windows by clicking X on the page, with "Close" button and so on. But there is another way that you can close the window, by pressing Alt+F4 on the keyboard. Not many

testers test this scenario. I have done this in my last two projects. Both the time, the application crashed which became a big issue. I felt proud.

42. What made you to choose testing career?

Answer: I am a very detailed oriented person and I like process-oriented job. The way QA process works is just the kind of work I like. For example, analyzing requirement documents, attending walk-through meetings, writing test plans, writing test cases, executing the test cases (or running the test cases) testing the application, logging defects, retesting them and so on. I think I really like the process and that's why I chose this career.

43. When should testing start in a project? Why?

Answer: We should start testing as soon as the following things are ready:

- Test Data are ready
- Build (all the developers have coded their code and merged them together)
- Test Environment (servers, network etc) is set up and ready
- When the manager asks us to go ahead and start testing.

44. Let us say you have a web application to test. How do you go about testing it? What is the process?

Answer: First of all, I will look at the requirement documents (or design document in some companies). The requirement document will tell us what the functionalities in the application (software) are. Once I analyze the requirement documents (one module=one requirement document). After that, I will write test plans for each module (one module =one test plan). Then after the test plan is complete, I will write test cases (One module can have hundreds, even thousands test cases). Once the test cases are ready and the application is ready (or once the build is ready), then I will start testing. Before I start testing, however, I will make sure the test environments, test data and defect logging tools are in place. This is how I will go about testing an application.

45. What is a "bug?"

Answer: A bug is a bug is an error, flaw, mistake, failure, or fault in a computer code (program) that prevents it from behaving as intended (e.g., producing an incorrect result). (You can also add this: When the expected results (accordingly to the requirement documents) don't match with the actual results (while testing), then it is considered a bug)

46. How would you ensure that you have covered 100% testing?

Answer: The testing coverage is defined by exit criteria (There is exit criteria and entry criteria in the Test Strategy). For example, if the exit criteria says "The software will be acceptable to the client only if there are no critical defects, no high defects, no medium defects and only two low defects", then all the critical, high, medium should be zero. Only 2 low defects are acceptable. Thus, 100% coverage is measured by the exit criteria. Also, 100% test cases must be executed in order to cover 100% of testing.

47. What problems did you face in the past? How did you solve it?

(You will be OK if you just give one of the problems below, not all of them)

Answer: I had many problems while testing applications in the past. As far as I remember one of them (then describe one of them from below), this was the scenario:

(i) It was a web-based application. I was working on a module called "Transaction Summary". There was "Submit" button on that page. After entering data in the all the fields, for example, First Name, Last Name, Social Security Number, Date of Birth and so on, I clicked the Submit button. Once I clicked Submit button, an error page displayed, "Page cannot be found...". Since it was a critical defect, I immediately informed the Test Lead. There was a chaos in the room. All the developers, Database Administrators and Testers gathered in my cube (room). No body could tell exactly what was wrong with it. Finally, one smart guy checked into the database and found out that one of the files in the database was closed. The status of all the files should be in

the open status. Once the status of the closed file was put in the "open" status, the application worked fine.

(ii) One of the problems was in the Login window (page). When the user enters and Login Name and Password, then Password should be encrypted. One of the Test Cases was that I needed to open database and see whether the password is encrypted or not. I found out it was not encrypted. I reported it as a bug (defect) and it was fixed in the next release (build).

(iii) Defects I have found in a project was a defect to close a window (pop up).

For example, in the last project, on a page, there was a button called "More Information". Once the user clicked that button, the system would open a new window (pop up). We could close the new window in 3 ways:

- By clicking X at the top right corner of the page

- By clicking "Close" button on the page

- By pressing combination keys (Alt+F4) on the key board

Although the combination key (Alt+F4) was not mentioned in the test case, I just wanted to try how the application reacts when Alt+F4 is pressed. Then I pressed Alt+F4. The result was a disaster-the application crashed (broke). The application disappeared from the computer monitor. Since it was the last day of testing for us, it brought chaos in our Managers, Leads and the whole teams. Finally, the developers disabled Alt+F4 as a temporary solution and the application went into production.

(iv) Another problem was that a user would search for branch location information of a bank. The user logs in by using User Name and Password. After the log in, on the "Search Location" page, the user enters and zip code of the location he wants to find, then clicks Find button. After that the system (application) gives a number of branch locations. The user now clicks "Request Information" for one of the branches. As soon as the user clicks "Request Information" button, the application breaks (displays "Page cannot be found" error). I logged this defect as a critical defect. When the developers and database administrator looked into it, then they found out that in one of the tables, the data was not recorded. In all the tables (UserProfile table, ClientID table and SessionID table), the data should be populated with the information entered by the user. For some reason, in one of the tables, it was blank (null). Once they wrote a small code to populate data (enter data) to the table, the application started working.

(v) In my previous project, when the customer wants to upload a document, for example, a copy of a monthly statement (in Word format), on the website, the system should automatically change the Word document into .pdf format. Once the document was uploaded, I saw that the fields in the .pdf document were interchanged (misplaced). For example, the First Name displayed in the Last Name section. Date of Birth displayed in the Social Security Number field and so on. We found out that the problem was a mapping problem (remember this word). Once the mapping was correct, I tested in the new build. It was fixed.

(vi) The most common problem that I have faced in my previous projects are the Java script errors, data connectivity, error, HTTP 500 error (This error occurs when server is down), HTTP 400 error (when file is not found) and so on.

(vii) "Father" pop up displayed when Print/Print Preview button clicked. (This was coded by the developer to mark this coding portion (for his/her own purpose as a mark to indicate where he/she made changes, however, forgot to remove it). Once the developer fixed it, it still displayed the same thing (because it was in the servers memory and could not go). Now, I had to reset memory of the server from my machine. Therefore, what I did is, I went to the website I was testing (for example, http://mysite.app.org/My_profile) and added reset.aspx at the end of the URL (Now the URL becomes http://mysite.app.org/My_profile/reset.aspx and hit enter. It took me to the server memory and I selected section and submitted the query and it was cleared. Retested again and it is now OK.

(viii) I was testing a web application. On one page, I clicked Save & Continue button twice (my mistake). Once this button is clicked twice, the system displayed an error message, "Could not save the answers, please contact technical support". (When clicked only once, the button works fine.).

Solution: Once the user clicks the button once, the button was disabled later so that the user cannot click twice.

(ix) I was testing a web-based application. Once all the fields are entered on the one of the pages, we had Print Preview button. If the user clicks this button, we were supposed see the same information in a new window in PDF format. While looking at the data in PDF file,

there were some fields missing, for example, Date of Birth was missing in the PDF file.

48. Tell me about the worst boss you've ever had.

(Here, you should be careful not to say any negative words about the past boss. This will give a reflection that you cannot work with different nature of people. You should be able to show them that you can cope with any king of boss. Therefore, just take an idea below how the answer should be.)

Answer: I can hardly think of any Manager that was really bad. But when I compare, then I remember of a Test Lead who was just made a lead from the developers team. She used to feel that she has been very proud of her position and used to boss around. Some times, she used to call home and check where I was and what I was doing. Or have I completed my job before leaving and so on. I think, whatever she did, was in the benefit of the company and myself in the long run which would give me more confidence in future.

49. What do you like about QA?

Answer: The best thing I like about QA is, I like the job which is more process oriented. For example, we have to work right from reading the requirement documents, providing feedback to the Business Analysts as necessary, writing test plans, test cases, execute the test cases, interaction with different developers, attend walk-through meeting and so on. I am a very detailed oriented person. When I test applications, I try to get into the depth of functionality so that I don't miss out anything. Finally, I love logging defects.

50. What are all the basic elements in a defect report?

Answer: The basic elements in a defect report are: Defect ID, Header, Description, Defect Reported by, Date, Status, Version, Assigned to, Approved by, Module where the defect was found and so on.

51. What is the difference between verification and validation?

Verification: Verification is a process to ensure that the software that is made, matches the original design. In other words, it checks

whether the software is made according to the criteria and specification described in the requirement document. It is to check whether you built the product right as per design. It is a low level checking. (It is done in walk-through meetings generally). It checked whether it is made accordingly to the design..

Validation: *Validation is a process to check whether the product design fits the client's need. It checks whether you built the right thing. It checks whether it is designed properly.*

52. How do you know it is sufficient testing?

Answer: Every company has entry and exit criteria. When we test applications, we refer to exit criteria. When we are about to finish testing, then the QA Team (QA Manager) refers to the exit criteria (exit criteria tells the level of defect that you can be comfortable with before it goes to production. For example, there should be ZERO critical defect, ZERO high level defect, ZERO medium defect, 1 Low level defect, all the test cases must be 100% executed etc). Once the exit criteria meet the requirements, then the software is considered to be sufficiently tested.

Every company has entry and exit criteria. When we test applications, we refer to exit criteria. When we are about to finish testing, then the QA Team (QA Manager) refers to the exit criteria (exit criteria tells the level of defect that you can be comfortable with before it goes to production. For example, there should be ZERO critical defect, ZERO high level defect, ZERO medium defect, 1 Low level defect, all the test cases must be 100% executed etc). Once the exit criteria meet the requirements, then the software is considered to be sufficiently tested.

53. How to derive test scenarios and use cases? What are the contents and format?

Answer: Test scenarios are derived from requirement documents. We follow each and every functionality (called business rules) mentioned in the requirement document. One functionality can have multiple business rules. For example, let us say in there is one requirement called "Login". This "Login" may have various scenarios. For example, one scenario is, enter the right User ID and wrong password. The system should display an error message. Another scenario would be to enter wrong User ID and right Password. The system should display an

error message. The third scenario could be to enter the right User Name and right Password. The system should allow the user to get into the system. This is how the test cases are derived from the requirement documents or from the Use Cases.

(For contents for formats of test scenario, please refer to question 4 in qaquestions.com)

54. What are the types of test cases that you write?

Answer: We write test cases for smoke testing, integration testing, functional testing, regression testing, load testing, stress testing, system testing and so on.

55. How to write Integration test cases?

Answer: I have never written separate Test Cases Integration Testing. Since Integration Testing is a test to check whether the all the modules are integrated together or not (meaning that when the developers compile all their module and make a build, all modules should be working when they are combined together and those modules when combined, should work as expected). If they are not integrated (combined) in a nice way, then the application breaks. Basically, when we do the functional testing, the integration testing is automatically done. This is my experience.

56. How to write Regression test cases? What are the criteria?

Answer: Regression test cases are also based on the requirement documents. They are written more into detail and with every release (build), the testers need to do regression testing. The criteria for regression testing are; there should be no major defects while we do our smoke test and functional testing.

57. Is there a format for a test case? Do you follow any methodology for numbering test cases?

Answer: Yes. It depends upon the company how the company has followed the numbering of test cases. However, normally, it is just a simple numbering in most of the time (see question 4 of qaquestions.com). But some companies may also relate this

numbering to the requirement number. For example, if the requirement for Login is "REQ-LOG-001", then we can number the test cases like REQ-LOG-001-001 and so on.

58. What is Test Harness?

Answer: (Definition from www.wikipedia.org) "In software testing, a test harness or automated test framework is a collection of software and test data configured to test a program unit by running it under varying conditions and monitor its behavior and outputs. It has two main parts: the test execution engine and the test script repository."

59. How to write User Acceptance Test plan & test cases?

Answer: The way of writing Test Plan and Test Cases is the same in all the test phases. However, specifically for User Acceptance Testing, the testers use data nearly real data (meaning that the data is very much similar to the production data or real data). For the format, please refer to question 3 and 4 in qaquestions.com.

60. What are the different matrices that you follow?

Answer: There are various reports we normally prepare in QA:

- Test summary Report – It is a report that has list of the total test cases, list of executed test cases, remaining test case to be executed, executed date, pass/fail
- Defect Report – In this report we normally prepare a list of defect in spreadsheet e.g. defect # CQ12345 [if you log a defect in the application called Rational ClearQuest]
- Traceability Matrix [also called RTM (Requirement Traceability Matrix)] Report – the document which shows the relationship between the functionalities or the business rules and the test cases. So, with the help of Traceability Matrix we make sure that we includes all the functionalities in our test cases according to the requirement document.

61. Explain Bug Life Cycle.

Answer: I would describe this as below:

A Tester finds a defect and logs it. (But before you log it, you must try

to recreate it for 3 or 4 times so that you are 100% sure that it is a bug)

The defect is now approved or disapproved by the Test Lead.

(If it is disapproved, then the test lead will come to you ask for more details and you have explain to him why it is a bug)

After the Test Lead approves the bug, it is now assigned to a development Team Lead (or Development Manager). He/she now assigns that bug to the concerned developer. The developer now looks into the bug and fixes it. Once the fix is ready, there will be another build ready to test. The tester now tests the defect. If the defect is fixed, then the tester closes the defect, if not then the test will reopen it and same cycle starts.

Defect Life Cycle

62. What will you do if developer does not accept the bug?

Answer: If the developer does not accept the defect, then he will reject it. Once it is rejected, then it comes back to the tester. Now, the tester will ask for clarification with the developer why the defect is rejected. Since everything is based on the requirement documents, both tester and developer will have to look at the requirement document, validate it and then reopen it if necessary or close.

63. What are the different tests that can be done for Client Server Application and Web-based Application. Give details.

Answer: For both client server and web based applications, the testing is the same except one thing: We test web based applications in different browsers, for example, Internet Explorer (will test in different versions like IE 5.0, IE 6.0, IE 7.0), Firefox, Safari (for Mac) and so on where as for client server, we don't need to test in the browsers.

64. What is an inspection?

Answer: An inspection is a formal meeting, more formalized than a walkthrough and typically consists of 3-10 people including a moderator, reader (the author of whatever is being reviewed) and a recorder (to make notes in the document). The subject of the inspection is typically a document, such as a requirements document or a test plan. The purpose of an inspection is to find problems and see what is missing, not to fix anything. The result of the meeting

should be documented in a written report. Attendees should prepare for this type of meeting by reading through the document, before the meeting starts; most problems are found during this preparation. Preparation for inspections is difficult, but is one of the most cost-effective methods of ensuring quality, since bug prevention is more cost effective than bug detection.

65. Give me five common problems that occur during software development.

Answer: Poorly written requirements, unrealistic schedules, inadequate testing, adding new features after development is underway and poor communication. Requirements are poorly written when requirements are unclear, incomplete, too general, or not testable; therefore there will be problems. The schedule is unrealistic if too much work is crammed in too little time.

Software testing is inadequate if none knows whether or not the software is any good until customers complain or the system crashes. It's extremely common that new features are added after development is underway.

Miscommunication either means the developers don't know what is needed, or customers have unrealistic expectations and therefore problems are guaranteed

66. What is the role of documentation in QA?

Answer: Documentation plays a critical role in QA. QA practices should be documented, so that they are repeatable. Specifications, designs, business rules, inspection reports, configurations, code changes, test plans, test cases, bug reports, user manuals should all be documented. Ideally, there should be a system for easily finding and obtaining of documents and determining what document will have a particular piece of information. Use documentation change management, if possible.

67. What if the software is so buggy it can't be tested at all?

Answer: In this situation the best bet is to have test engineers go through the process of reporting whatever bugs or problems initially show up, with the focus being on critical bugs. Since this type of

problem can severely affect schedules and indicates deeper problems in the software development process, such as insufficient unit testing, insufficient integration testing, poor design, improper build or release procedures, managers should be notified and provided with some documentation as evidence of the problem.

68. How do you know when to stop testing?

Answer: This can be difficult to determine. Many modern software applications are so complex and run in such an interdependent environment, that complete testing can never be done. Common factors in deciding when to stop are...

*Deadlines, e.g. release deadlines, testing deadlines;
Test cases completed with certain percentage passed;
Test budget has been depleted;
Coverage of code, functionality, or requirements reaches a specified point;
Bug rate falls below a certain level; or
Beta or alpha testing period ends.*

69. What if there isn't enough time for thorough testing?

Answer: Since it's rarely possible to test every possible aspect of an application, every possible combination of events, every dependency, or everything that could go wrong, risk analysis is appropriate to most software development projects. Use risk analysis to determine where testing should be focused. This requires judgment skills, common sense and experience. The checklist should include answers to the following questions:

- Which functionality is most important to the project's intended purpose?*
- Which functionality is most visible to the user?*
- Which functionality has the largest safety impact?*
- Which functionality has the largest financial impact on users?*
- Which aspects of the application are most important to the customer?*
- Which aspects of the application can be tested early in the development cycle?*
- Which parts of the code are most complex and thus most subject to errors?*

- Which parts of the application were developed in rush or panic mode?
- Which aspects of similar/related previous projects caused problems?
- Which aspects of similar/related previous projects had large maintenance expenses?
- Which parts of the requirements and design are unclear or poorly thought out?
- What do the developers think are the highest-risk aspects of the application?
- What kinds of problems would cause the worst publicity?
- What kinds of problems would cause the most customer service complaints?
- What kinds of tests could easily cover multiple functionalities?
- Which tests will have the best high-risk-coverage to time-required ratio?

70. What can be done if requirements are changing continuously?

Answer: Work with management early on to understand how requirements might change, so that alternate test plans and strategies can be worked out in advance. It is helpful if the application's initial design allows for some adaptability, so that later changes do not require redoing the application from scratch. Additionally, try to...
· Ensure the code is well commented and well documented; this makes changes easier for the developers.

- Use rapid prototyping whenever possible; this will help customers feel sure of their requirements and minimize changes.
- In the project's initial schedule, allow for some extra time to commensurate with probable changes.
- Move new requirements to a 'Phase 2' version of an application and use the original requirements for the 'Phase 1' version.
- Negotiate to allow only easily implemented new requirements into the project; move more difficult, new requirements into future versions of the application.
- Ensure customers and management understand scheduling impacts,

inherent risks and costs of significant requirements changes. Then let management or the customers decide if the changes are warranted; after all, that's their job.

- Balance the effort put into setting up automated testing with the expected effort required to redo them to deal with changes.
- Design some flexibility into automated test scripts;
- Focus initial automated testing on application aspects that are most likely to remain unchanged;
- Devote appropriate effort to risk analysis of changes, in order to minimize regression-testing needs;
- Design some flexibility into test cases; this is not easily done; the best bet is to minimize the detail in the test cases, or set up only higher-level generic-type test plans;
- Focus less on detailed test plans and test cases and more on ad-hoc testing with an understanding of the added risk this entails.

71. What if the application has functionality that wasn't in the requirements?

Answer: It may take serious effort to determine if an application has significant unexpected or hidden functionality, which it would indicate deeper problems in the software development process. If the functionality isn't necessary to the purpose of the application, it should be removed, as it may have unknown impacts or dependencies that were not taken into account by the designer or the customer. If not removed, design information will be needed to determine added testing needs or regression testing needs. Management should be made aware of any significant added risks as a result of the unexpected functionality. If the functionality only affects areas, such as minor improvements in the user interface, it may not be a significant risk.

72. How can software QA processes be implemented without stifling productivity?

Answer: Implement QA processes slowly over time. Use consensus to

reach agreement on processes and adjust and experiment as an organization grows and matures. Productivity will be improved instead of stifled. Problem prevention will lessen the need for problem detection. Panics and burnout will decrease and there will be improved focus and less wasted effort. At the same time, attempts should be made to keep processes simple and efficient, minimize paperwork, promote computer-based processes and automated tracking and reporting, minimize time required in meetings and promote training as part of the QA process. However, no one, especially talented technical types, like bureaucracy and in the short run things may slow down a bit. A typical scenario would be that more days of planning and development will be needed, but less time will be required for late-night bug fixing and calming of irate customers.

73. What is parallel/audit testing?

Answer: Parallel/audit testing is testing where the user reconciles the output of the new system to the output of the current system to verify the new system performs the operations correctly. Let us say, for example, the currently software is in the mainframe system which calculates the interest rate. The company wants to change this mainframe system to web-based application. While testing the new web based application, we need to verify that the web-based application calculates the same interest rate. This is parallel testing.

74. What is system testing?

Answer: System testing is black box testing, performed by the Test Team, and at the start of the system testing the complete system is configured in a controlled environment. The purpose of system testing is to validate an application's accuracy and completeness in performing the functions as designed. System testing simulates real life scenarios that occur in a "simulated real life" test environment and test all functions of the system that are required in real life. System testing is deemed complete when actual results and expected results are either in line or differences are explainable or acceptable, based on client input.

Upon completion of integration testing, system testing is started. Before system testing, all unit and integration test results are reviewed by Software QA to ensure all problems have been resolved.

For a higher level of testing it is important to understand unresolved problems that originate at unit and integration test levels. You CAN learn system testing, with little or no outside help. Get CAN get free information. Click on a link!

75. What is end-to-end testing?

Answer: Similar to system testing, the *macro* end of the test scale is testing a complete application in a situation that mimics real world use, such as interacting with a database, using network communication, or interacting with other hardware, application, or system.

76. What is security/penetration testing?

Answer: Security/penetration testing is testing how well the system is protected against unauthorized internal or external access, or willful damage. This type of testing usually requires sophisticated testing techniques.

77. What is recovery/error testing?

Answer: Recovery/error testing is testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.

78. What is compatibility testing?

Answer: Compatibility testing is testing how well software performs in a particular hardware, software, operating system, or network environment.

79. What is comparison testing?

Answer: Comparison testing is testing that compares software weaknesses and strengths to those of competitors' products.

80. What is acceptance testing?

Answer: Acceptance testing is black box testing that gives the client/customer/project manager the opportunity to verify the system functionality and usability prior to the system being released to

production. The acceptance test is the responsibility of the client/customer or project manager, however, it is conducted with the full support of the project team. The test team also works with the client/customer/project manager to develop the acceptance criteria.

81. What is a Test/QA Team Lead?

Answer: The Test/QA Team Lead coordinates the testing activity, communicates testing status to management and manages the test team.

82. What is software testing methodology?

Answer: One software testing methodology is the use a three step process of...

1. Creating a test strategy;
2. Creating a test plan/design; and
3. Executing tests. This methodology can be used and molded to your organization's needs. Rob Davis believes that using this methodology is important in the development and in ongoing maintenance of his customers' applications.

83. What is the general testing process?

Answer: The general testing process is the creation of a test strategy (which sometimes includes the creation of test cases), creation of a test plan/design (which usually includes test cases and test procedures) and the execution of tests.

84. How do you create a test strategy?

Answer: The test strategy is a formal description of how a software product will be tested. A test strategy is developed for all levels of testing, as required. The test team analyzes the requirements, writes the test strategy and reviews the plan with the project team. The test plan may include test cases, conditions, the test environment, a list of related tasks, pass/fail criteria and risk assessment. Inputs for this process:

- A description of the required hardware and software components, including test tools. This information comes from the test

environment, including test tool data.

- A description of roles and responsibilities of the resources required for the test and schedule constraints. This information comes from man-hours and schedules.
- Testing methodology. This is based on known standards.
- Functional and technical requirements of the application. This information comes from requirements, change request, technical and functional design documents.
- Requirements that the system can not provide, e.g. system limitations. Outputs for this process:
- An approved and signed off test strategy document, test plan, including test cases.
- Testing issues requiring resolution. Usually this requires additional negotiation at the project management level.

85. How do you create a test plan/design?

Answer: Test scenarios and/or cases are prepared by reviewing functional requirements of the release and preparing logical groups of functions that can be further broken into test procedures. Test procedures define test conditions, data to be used for testing and expected results, including database updates, file outputs, report results. Generally speaking...

Test cases and scenarios are designed to represent both typical and unusual situations that may occur in the application.

Test engineers define unit test requirements and unit test cases. Test engineers also execute unit test cases.

It is the test team that, with assistance of developers and clients, develops test cases and scenarios for integration and system testing.

Test scenarios are executed through the use of test procedures or scripts.

Test procedures or scripts define a series of steps necessary to perform one or more test scenarios.

Test procedures or scripts include the specific data that will be used for testing the process or transaction.

Test procedures or scripts may cover multiple test scenarios.

Test scripts are mapped back to the requirements and traceability matrices are used to ensure each test is within scope.

Test data is captured and base lined, prior to testing. This data serves as the foundation for unit and system testing and used to exercise

system functionality in a controlled environment.

Some output data is also base-lined for future comparison. Base-lined data is used to support future application maintenance via regression testing.

A pretest meeting is held to assess the readiness of the application and the environment and data to be tested. A test readiness document is created to indicate the status of the entrance criteria of the release.

Inputs for this process:

Approved Test Strategy Document.

Test tools, or automated test tools, if applicable.

Previously developed scripts, if applicable.

Test documentation problems uncovered as a result of testing.

A good understanding of software complexity and module path coverage, derived from general and detailed design documents, e.g. software design document, source code and software complexity data.

Outputs for this process:

Approved documents of test scenarios, test cases, test conditions and test data.

Reports of software design issues, given to software developers for correction.

86. How do you execute tests?

Answer: Execution of tests is completed by following the test documents in a methodical manner. As each test procedure is performed, an entry is recorded in a test execution log to note the execution of the procedure and whether or not the test procedure uncovered any defects. Checkpoint meetings are held throughout the execution phase. Checkpoint meetings are held daily, if required, to address and discuss testing issues, status and activities. The output from the execution of test procedures is known as test results. Test results are evaluated by test engineers to determine whether the expected results have been obtained. All discrepancies/anomalies are logged and discussed with the software team lead, hardware test lead, programmers, software engineers and documented for further investigation and resolution. Every company has a different process for logging and reporting bugs/defects uncovered during testing. A pass/fail criteria is used to determine the severity of a problem, and results are recorded in a test summary report. The severity of a problem, found during system testing, is defined in accordance to the

customer's risk assessment and recorded in their selected tracking tool. Proposed fixes are delivered to the testing environment, based on the severity of the problem. Fixes are regression tested and flawless fixes are migrated to a new baseline. Following completion of the test, members of the test team prepare a summary report. The summary report is reviewed by the Project Manager, Software QA Manager and/or Test Team Lead.

After a particular level of testing has been certified, it is the responsibility of the Configuration Manager to coordinate the migration of the release software components to the next test level, as documented in the Configuration Management Plan. The software is only migrated to the production environment after the Project Manager's formal acceptance.

87. What testing approaches can you tell me about?

Answer: Each of the followings represents a different testing approach:

Black box testing,
White box testing,
Unit testing,
Incremental testing,
Integration testing,
Functional testing,
System testing,
End-to-end testing,
Sanity testing,
Regression testing,
Acceptance testing,
Load testing,
Performance testing,
Usability testing,
Install/uninstall testing,
Recovery testing,
Security testing,
Compatibility testing,
Exploratory testing, ad-hoc testing,
User acceptance testing,
Comparison testing,
Alpha testing,

Beta testing, and
Mutation testing.

88. How do you divide the application into different sections to create scripts?

Answer: First of all, the application is divided in different parts when a business analyst writes the requirement document (or Use Cases or Design Document), he/she writes EACH requirement document for EACH module. Let us say, if there are 12 different modules in an application that a business analyst has written the requirements for, then a tester would write the test cases for each module, which means in 12 different sections. This is the standard practice. There might be scenarios where you might have to break down scripts into sub-categories. For example, if a tester is writing a script for Login Page, he/she might write one for positive and negative testing and another sub-set of test cases would be for error message when the wrong information is entered. In short, the test cases are divided according to the modules.

(The following questions were asked to Padma in one of her interviews very recently)

(This question is asked to check how ambitious you are as far as your career is concerned, whether you like the job you are doing and so on. Therefore, no matter what, you should stick to your QA job at this point and say that you love this so much and your goal is some thing similar to the one below)

What is your salary requirement?

\$70k (negotiable), or (\$35 per hour)

Please provide information (an example) of your experience testing Linux and UNIX environments (including type of system tested, how tested, actual commands and steps used for test) Testing applications using Linux and UNIX.

Answer: I have tested applications using UNIX. For every backend testing I have done in the past, I have used UNIX platform while performing backend testing. For example, when the data is fed into the system in the front end, that data goes to the database after the batch processing. From the database, the data is now sent to the ETL system (in XML format) for data manipulation as per our need (ETL is

a software tool of Ab Initio company which is used to manipulate data in the data warehouse). In the ETL system, we manipulate those data according to our need), for example, it could be income statement of the company, balance sheet, monthly reports, and so on. In order to produce income statement, we need to run a job in ETL. To run this job, we use UNIX. In the same way, different types of jobs are created for each need (creating balance sheet is another job, creating reports is next job etc) then I had to run different jobs in the ETL system. Once we run the job, the running job finally creates an output file which is now validated by us tester. This output file can be in text format or GUI format. Thus, this is the scenario where I had to use UNIX. (I have used Linux much, however, since UNIX and Linux are the same thing, I should have no problem in using Linux) Some of the commands I used while testing using UNIX are;

Ls -l ---->to check the file list
 Pwd-----> to see which directory I am in
 Cd ----->change the directory
 Cd .. ---->change the directory one level up
 Mkdir ---->make a directory
 Rmdir ---->Delete the directory
 setenv name v --->Set environment
 kill% ---->Kill the running job
 vi ---->editor Used to write scripts
 more----> to see the contents page by page
 cat --->list contents of the file
 chmod ---->change permission
 cp ---->copy
 rm --->delete a file

The following are the some of the things that a tester has to know (but may not be asked in the interview)What is a cookie? (You must know how to clean cookies)

A small text file of information that certain Web sites attach to a user's hard drive while the user is browsing the Web site. A Cookie can contain information such as user ID, user preferences, archive shopping cart information, etc. Cookies can contain Personally Identifiable Information.

Does a tester have to know about cookie?

Yes. A tester has to know HOW TO CLEAN cookies (Does not have to know the difinition)

How to clean cookies?

Cookies are cleaned in the browsers like IE (Internet Explorer), Firefox, Safari (for MAC and windows both), Netscape and so on.

However, the mostly used (90%) browser is IE (Internet Explorer)

Here is how you clean cookies in IE (Internet Explorer):

1. Open IE (Internet Explorer)
2. On the menu, click Tools->Internet Options->Click Delete button (It is in General Tab)
(You will see different buttons now, for example, Delete Files, Delete Cookies, Delete History, Delete Forms, Delete Passwords, Delete All).

3. Click Delete All button.

Now the cookies are cleaned in IE.

Here is how you can clean cookies in Fire Fox:

1. Open Firefox Brower.
2. Click Tools.
3. Click Error Console.
4. Click Clear.

Now the cookies are cleaned in Firefox.

What are different types of protocols?

-Generally, a Tester does NOT necessarily have to know different types of protocols. This is Network Engineers job. However, if you want to know more for your knowledge, you can visit: http://en.wikipedia.org/wiki/List_of_network_protocols.

What is Web Architecture?

-A tester does not necessarily have to know this unless you are a very Senior Tester testing networks and doing some kind of development. However, if you want to know more about it, please visit: <http://www.objs.com/survey/WebArch.htm>

Does a Tester need SQL?

Answer: Yes. For a Tester, SQL is needed. I had the same question in mind before I came to the actual implication-what is SQL used for? And now, I know that when we do the backend testing (see

qaquestions.com for details), we need to write SQL queries to retrieve the data from the database and compare this data to the one with reports or output. Another scenario is, if something goes wrong in the application, for example, if there is an error, then we might have to write SQL queries to retrieve the data from the database and check what went wrong. Let's say, we need to check in the Error Log table what went wrong. To check this, we open the database, go to Error Log table and find out that happened. In the Error Log table, there are many records, so which one is your error then? To find out which one is yours, we need to write SQL queries. Example, you logged in to the application with User and password=sn992jj. Now, to retrieve your record, you can write a query some thing like this: `select * from Error_Log where userID=devin99;` This query will retrieve your record only so that you can see what happened.

What is a 'Show Stopper'?

A show stopper is a defect or bug that stops the user for further action (testing). It has no work around. In other words, it stops everything and the user cannot go any further. This is called show stopper in software industry language. (This is not an interview question, but you have to know this terminology)

Some Glossary

Test Plan, Test Case, Test Script, Requirement Document, Design Document, Shared Drive, Network Drive, Share Point, System, Build Configuration Management Team, Defect, Log, Automation Tools, TestDirector, Quality Center, ClearQuest, ClearCase, Rational Robot, Rational Functional Tester, WinRunner, LoadRunner, Business Objects, Crystal Reports, SQA, QA

Answer: My goal is to be QA Lead (or QA Manager) in near future.

90. What are you expecting from our company?

Answer: My expectation from your company would be I will have more challenges and new things to learn and whatever the skills I have to contribute, hopefully, I will be able to contribute if they are in any way helpful to enhance productivity of the company.

91. What did you learn from your previous companies?

Answer: I learned a lot from the previous companies wherever I have

worked. Wherever I have worked, I found out that there is always something to learn. Different companies have different ways of working. The environment and technology always differ from one company to another company. I have never found one company's environment matching with another company. For example, if one company is using documents called requirement documents, then the other company might be using Use Cases and some companies might be using Design Document and so on. Therefore, in my experience, there are always new things to learn in every company and we can always contribute these things in the next company if they help to be more productive.

92. What do you want to be in next 2 years?

Answer: I want to be QA Lead in another two years.

Why QA Lead? Why not something else?

Answer: QA is the only thing I love doing it. I love this job and want to progress in this sector. I want to know how to manage QA process, how to handle different jobs and so on. Since the next step is the QA Lead, that would preferably be one I will targeting for.

93. Why do you want to work for this company?

Answer: (This is a tricky question. They want to know what really interests you and you have to be careful when you answer this question. You must admire the line of that company. For example, if you are being interviewed by a pharmaceutical company, then tell them that you are always interested in the medical applications and the better part of your company is that it has exciting products that I am really curious to learn. That's why I would feel really great if I am given the opportunity to work in your company)

94. Did you get any compliments from your previous employers? What were those situations?

Answer: Yes. I did. There were many occasions where I had compliments. For example, I was testing an application going a little bit off my test cases. After I finished executing my test cases, I always think in a way what a real user would possibly click in various parts

of the application. So I was just clicking back and forth and at one specific scenario, the application simply broke and displayed an error message. That scenario was not in the test cases. The manager really appreciated me and thanked for finding this kind of critical defect. Answer: Yes. I did. There were many occasions where I had compliments. For example, I was testing an application going a little bit off my test cases. After I finished executing my test cases, I always think in a way what a real user would possibly click in various parts of the application. So I was just clicking back and forth and at one specific scenario, the application simply broke and displayed an error message. That scenario was not in the test cases. The manager really appreciated me and thanked for finding this kind of critical defect.

What are your strengths?

Answer: I am a very detailed oriented person. I have the sense of urgency. I can prioritize my job according to the deadline. I am very much dedicated towards my job. I am honest. I have the skills and expertise in QA process. These are some of my strengths.

What is your weakness?

Answer: I think my weakness is that whenever I am given some responsibilities and there is a deadline for it, I work day and night, 7 days a week. This is probably bad for my family life, but I can't sleep unless I am done with my assignments.

(Note: You should think of your weakness where because of your weakness (like the one above), still the employer benefits. DON'T SAY anything negative thing, like "I cannot work long hours, it is hard for me pick up things, it is difficult for me to understand requirement documents etc)

89. What is your goal?

Ans: My goal in the next 4 years is to be a QA Manager.

90. What is RTM (Requirement Traceability Matrix)?

Answer – Tractability matrix is used to cross check the test cases as per the requirement of the test cases. In other words, it checks whether the each functionality is covered in the Test Cases as per requirement document. (We create RTM using Quality Center tool)

- Category: [Software Testing](#)

1).What is WebServices?

It is a middleware developed in XML. Exchange the data between multiple platforms.

2).What is web services?

It is middleware technology developed in XML to exchange the data between multiple platforms and languages.

3).what is rest protocol ?

REST stands for Representational State Transfer. (It is sometimes spelled "ReST".) EST is a lightweight alternative to mechanisms like RPC (Remote Procedure Calls) and Web Services (SOAP, WSDL, et al.). Later, we will see how much more simple REST is.

Despite being simple, REST is fully-featured; there's basically nothing you can do in Web Services that can't be done with a RESTful architecture.

REST is not a "standard". There will never be a W3C recommendation for REST, for example. And while there are REST programming frameworks, working with REST is so simple that you can often "roll your own" with standard library features in languages like Perl, Java, or C#.

4).In Interviews they ask do you have work experience in web based applications, does that mean having experience in web services in SOAPUI ?

Ans: NO

5). What is synchronous web services? -- get response immediately.

6). What is asynchronous web services? --Response will be sent when the service is available.

7). What is SOAPUI? It is a tool to test the WebServices.

8). what is xml schema?-- XML schema is well defined xml structure. It is w3 standards.

9). In Interviews if they ask do you have work experience in web based applications, does that mean having experience in web services in soapUI?

Ans: NO.

10).If any application accessed through internet/ using browser , that is web based application?

Not every Web based application required web services.

Now a day's most of the web based applications are using WebServices.

11). How do we know the project has WebServices?

Ans: You need to know the application architecture which is explained by your team. You need to login to the servers and see if any wsdl files, WebServices running on the server where the application is deployed.

Check for the process running the server.

12). What is end point in WebServices?

Ans : End point is IP Address of the server where the WebServices are running. for example: if the WebServices are running on the Linux machine

ip address: 145.12.4.5

Port number: 8080

WebServices WSDL name is: ATTBilling

- [6 Comments](#) – [Feed](#)

•

- **Navigation**

- [Nature of Work in Software Industry](#)
- [Before Other Documents- Read This](#)
- [DOCUMENTS REFERRED BY \(DEVELOPERS/TESTERS\)](#)
 - [1. Business Requirement Document \(BRD\)](#)
 - [2.Design Document \(Sample\)](#)
 - [3.Use Cases \(Sample\)](#)
- [*****](#)
- [What happens when the requirements are finalized?](#)
- [LIFE CYCLE](#)
 - [Software Development Life Cycle \(SDLC\) vs. Software Testing Life Cycle \(STLC\)-Diagram](#)
 - [Software Testing Life Cycle \(STLC\)](#)
- [SAMPLE DOCUMENTS \(For Testers Only\)](#)
 - [—Test Strategy \(Sample\)](#)
 - [—Test Plan \(Sample\)](#)
 - [—Test Cases \(Sample\)](#)

***** OTHER TESTING RELATED QUESTIONS

AVAILABLE TRAINING

- Manual Testing
 - Interview Preparation
 - Complete QA Training
 - SQL Basic
 - UNIX Basic
- Automation Testing Tools
 - Quality Center
 - (QTP) Quick Test Pro
 - LoadRunner

Categories

QA Interview Questions and Answers
Software Testing

Archives

December 2009

Blogroll

Development Blog (LoadRunner Tutorial)
Documentation (Put this in the right place)
Plugins
Suggest Ideas
Support Forum
Themes

Syndicate

Articles Feed
Comments Feed

Meta

Register
Log in

Interview Questions for QA Tester

- [WordPress](#) | [Simplify](#)

Q. 1: Why a software needs to be tested?

Every software product needs to be tested since, the development 'process' is unable to produce defect free software. Even if the development process is able to produce a defect free software, we will not be able to know unless & until we test it. Without testing it, we shall not be having enough confidence that it will work.

Testing not only identifies and reports defect but also measures the quality of the product, which helps to decide whether to release the product, or not.

<<<<< ===== >>>>>

Q. 2: What is the reason that Software have Bugs?

Following factors contribute to the presence of bugs in the software applications.

- 1) Software development tools like visual tools, class libraries, compilers, scripting tools, etc. usually introduce their own bugs in the system.
- 2) To err is human. Likewise programmers do make mistakes while programming.
- 3) In fast-changing business environments continuously modified requirements are becoming a fact of life. Such frequent changes requested by the customer leads to errors in the application already nearing completion. Last minute design changes leads to many chaos like redesign of the whole system, rescheduling of engineers, scrapping of the work already completed, fresh requirements of compatible hardware etc.
- 4) A quickly written but poorly documented code is bound to have bugs. It becomes difficult to maintain and modify such code that is badly written or poorly documented.
 - it's tough to maintain and modify code that is badly written or poorly documented; the result is bugs. In many organizations management provides no incentive for programmers to document their code or write clear, understandable, maintainable code. In fact, it's usually the opposite: they get points mostly for quickly turning out code, and there's job security if nobody else can understand it ('if it was hard to write, it should be hard to read').
- 5) When project deadlines come too close & time pressures come, mistakes are bound to come.

<<<<< ===== >>>>>

Q. 3: What is the difference between QA and Testing?

QA stands for "Quality Assurance", and focuses on "Prevention" of defects in the product being developed. It is associated with the "Process" and activities related to the Process

Improvement. Quality Assurance measures the quality of the processes employed to create a quality product.

Whereas "Testing" refers to "Quality Control", and focuses on Detection of Defect and removal thereafter. Or Quality Control measures the quality of a product.

<<<<< ===== >>>>>

Q. 4: What is the difference between Software Testing and Debugging?

Testing is the process of locating or identifying the errors or bugs in a software system.

Whereas Debugging is the process of Fixing the identified Bugs. It involves a process of analyzing and rectifying the syntax errors, logic errors and all other types of errors identified during the process of testing.

<<<<< ===== >>>>>

Q. 5: What is the difference between a Bug and a Defect?

"Bug" is a problem or an error in the software code, which is found in the application during Testing. Bug is responsible for failure of the application to comply with the desired specifications.

Whereas "Defect" is problem reported by the customer during usage of the software application.

<<<<< ===== >>>>>

Q. 6: What is the difference between a Bug and an Enhancement?

"Bug" is a problem or an error in the software code, which is found in the application during Testing. Bug is responsible for failure of the application to comply with the desired specifications.

Whereas "Enhancement" is the additional feature or functionality found and added to the application as desired by the end user / real word customer or tester during the testing process.

<<<<< ===== >>>>>

Q. 7: What is the difference between Requirements & Specifications?

"Requirements" are statements given by the customer as to what needs to be achieved by the software system. Later on these requirements are converted into specifications which are nothing but feasible or implementable requirements.

Whereas "Specifications" are feasible requirements derived from various statements given by the customer. These are the starting point for the product development team.

<<<<< ===== >>>>>

Q. 8: What is the sequence of succession in STLC - Software Testing Life Cycle?

- 1) Test Planning
- 2) Test Analysis
- 3) Test Design
- 4) Construction and verification
- 5) Testing Cycles
- 6) Final Testing and Implementation and Post Implementation.

<<<<< ===== >>>>>

Q. 9: What is the difference between Verification and Validation?

"Verification" involves reviews and meetings to evaluate documents, plans, code, requirements, and specifications to confirm whether items, processes, services, or documents conform to specified requirements or not. This can be done with the help of checklists, issues lists, walkthroughs, and inspection meetings. The purpose of verification is to determine whether the products of a given phase of the software development cycle fulfill the requirements established during the previous phase or not.

Whereas "Validation" is the determination of the correctness of the final program or software product produced from a development project with respect to the user needs and requirements. This involves actual testing of the product and takes place after verifications are completed.

"Software Verification" raises the question, "Are we building the Product Right?"; that is, does the software conform to its specification.

"Software Validation" raises the question, "Are we building the Right Product?"; that is, is the software doing what the user really requires.

<<<<< ===== >>>>>

Q. 10: What is the difference between a Test Plan and a Use Case?

"Test Plan" is a document describing an introduction to the client company, intended scope, overview of the application, test strategy, schedule of testing activities, roles and responsibilities, deliverables and milestones. It describes test items, features to be tested, testing tasks, details of the personnel performing each task and any risks requiring contingency planning.

Whereas a "Use Case" describes the process as to how an end user uses a specific functionality in the application. It is a summary of user actions and system response to the user actions. It contains the flows like typical flow, alternate flow and exceptional flow. It also contains pre condition and post condition.

<<<<< ===== >>>>>

Q. 11: What is the difference between Bug Priority & Bug Severity?

"Bug Priority" is the need on how urgently bug is needed to be fixed. It describes the importance of the bug. Bug priority may change according to the schedule of testing.

Whereas "Bug Severity" is the quantum of danger as to how badly the bug can harm the system. It describes as to how bad the bug is. Severity is a feature of constant nature associated with the bug.

<<<<< ===== >>>>>

Q. 12: What is difference between Waterfall Model and V Model?

"Waterfall Model" Is a sequential software development model (a process for the creation of software) in which development is seen as flowing steadily downwards (like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance. To follow the waterfall model, we proceed from one phase to the next in a purely sequential manner. In traditional waterfall model, testing comes at the fag end of the development process.

Whereas "V Model" or "Life Cycle Testing" involves carrying out verification of consistency, completeness and correctness of software at every stage of the development life cycle. It aims at catching the defects as early as possible and thus reduces the cost of fixing them. It involves continuously testing the system during all stages of the development process rather than just limiting testing to the last stage.

<<<<< ===== >>>>>

Q. 13: What are Baseline Documents?

Baseline documents are the documents, which have been approved by the customer and will not have any more changes. Baseline Documents cover all the details of the project and have undergone "walkthrough" process. Once a document is Base-lined it cannot be changed unless there is a change request duly approved by the customer. Service Level Agreement (SLA) & Business Requirement Documents (BRD) are the examples of Baseline Documents.

<<<<< ===== >>>>>

Q. 14: What is Defect Density?

"Defect Density" Is a software metric defined as: Total number of defects per LOC (lines of code). Alternatively It can be: Total number of defects per Size of the Project. Here the measure of "Size of the Project" can be number of Function Points, Number of Feature Points, number of Use Cases or KLOC (Kilo Lines of Code) etc

<<<<< ===== >>>>>

Q. 15: What is Negative Testing?

"Negative Testing" involves testing the application for failure like conditions. It involves testing the tool with improper inputs. For example entering the special characters in place of a phone number.

<<<<< ===== >>>>>

Q. 16: What is Incremental Integration Testing?

"Incremental Integration Testing" Involves continuous testing of an application while new functionality is simultaneously added. It requires that various aspects of an application's functionality be independent enough to work separately before all parts of the program are completed. This testing is done either by programmers or by testers.

<<<<< ===== >>>>>

Q. 17: What is the difference between Unit Testing, Component Testing and Integration Testing?

"Unit Testing" involves testing of individual programs, modules, or components to demonstrate that the program executes as per the specification and it validates the design and technical quality of the application. In Unit Testing, the Called Components (or Communicating Components) are replaced with Stubs, Simulators, or Trusted Components. Testing Stubs or Drivers are used to simulate the behavior of interfacing modules.

"Component Testing" is like "Unit Testing" with the difference that all Stubs and Simulators are replaced with the real objects. Here a Unit is a component, and integration of one or more such components is also a Component.

Whereas "Integration Testing" is the test process which begins after two or more programs components have been successfully unit tested. It is conducted by the development team to validate the interaction or communication/flow of information between the individual components that will be integrated.

<<<<< ===== >>>>>

Q. 18: What is the difference between Statement Coverage, Branch Coverage and Path Coverage?

"Statement Coverage" is a type of "White-Box Testing" technique, involving execution of all statements at least once. Statement coverage is a simple metric to calculate & measure the number of statements in a method or class which have been executed. Its key benefit is its ability to identify which blocks of code have not been executed.

"Branch Coverage" is an outcome of a decision, and measures the number of decision outcomes or branches, which have been tested. This takes a more in-depth view of the source code rather than a simple "Statement Coverage". A branch is an outcome of a decision. For example Boolean decisions like an "If - Statement", has two outcomes or branches (i.e. True and False).

Whereas "Path Coverage" is a method of testing which satisfies the coverage criteria

through which the program is tested across each logical path. Usually, paths through the program are grouped into a finite set of classes and one path out of every class is tested. In Path Coverage flow of execution takes place from the start of a method to its exit. Path Coverage ensures that we test all decision outcomes independently of one another.

<<<<< ===== >>>>>

Q. 19: What is the difference between Ad-hoc Testing, Monkey Testing and Exploratory Testing?

"Ad-hoc Testing" is performed without any planning of process and without any documentation like Test Case or Test Scenarios. It involves test design and simultaneous test execution. For Ad-hoc testing the testers possess significant understanding of the software before testing it.

"Monkey Testing" is done with no specific test in mind. Here the monkey is the producer of any input data (which can be either a file data or can be an input device data). It involves pressing some keys randomly and checking whether the software fails or not.

Whereas "Exploratory Testing" involves simultaneous learning, test design and test execution. It is a type of "Ad-hoc Testing", but only difference is that in this case, the tester does not have much idea about the application & he explores the system in an attempt to learn the application and simultaneously test it.

<<<<< ===== >>>>>

Q. 20: What is the difference between System Testing and End-to-End Testing or E2E Testing?

"System Testing" falls within the scope of Black-Box testing and the tester requires no knowledge of the inner design of the code or logic. It is conducted on a complete / combined part of a system to verify that all-functional, information, structural and quality requirements as per the specifications have been met.

"End-to-End Testing" or "E2E Testing" is also quite similar to "System Testing". It involves testing of the application in a environment that simulates the real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems.

<<<<< ===== >>>>>

Q. 21: What is the difference between Alpha Testing and Beta Testing?

Typically a software product passes through two stages of testing before it is considered to be Final. The first stage is known as "Alpha Testing". It is often performed by potential users / customers or an independent test team at the developers' site. It is usually done when the development of the software product is nearing completion; minor design changes may still be made as a result of Alpha testing.

The second stage coming after alpha testing is known as "Beta Testing". Versions of the

software, known as beta versions, are released to a limited audience outside of the programming team so that further evaluation by the users can reveal more faults or bugs in the product. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximum number of future users.

<<<<< ===== >>>>>

Q. 22: What is the difference between Static Testing and Dynamic Testing?

"Static Testing" involves testing activities performed without actually running the software. It includes Document review, code inspections, walkthroughs and desk checks etc.

Whereas "Dynamic Testing" Is used to describe the testing of the dynamic behavior of the software code. It involves actual compilation & running of the software by giving input values and checking if the output is as expected. It is the validation portion of Verification and Validation.

<<<<< ===== >>>>>

Q. 23: What is the difference between Smoke Testing and Sanity Testing?

The general term of "Smoke Testing" has come from leakage testing of sewers & drain lines involving blowing smoke into various parts of the sewer and drain lines to detect sources of unwanted leaks and sources of sewer odors. In relation to software testing field, Smoke testing Is a non-exhaustive software testing, ascertaining that the most crucial functions of the program work well, without getting bothered about finer details of it.

Whereas "Sanity Testing" Is an initial testing effort to find out if the new software version is performing well enough to accept it for a major testing effort. For example, if the new software is crashing the systems every 5 minutes, bogging down the systems to a crawl, or destroying the databases, then it can be concluded that the software may not be in a 'sane' enough condition to warrant further testing in its current state.

<<<<< ===== >>>>>

Q. 24: What is the difference between Stress Testing and Load Testing?

"Stress Testing" is subjecting a system to an unreasonable load while denying it the adequate resources as required to process that load. The resources can be RAM, disc space, mips & interrupts etc. etc. The idea is to stress a system to the breaking point in order to find bugs, which will make the break potentially harmful. The system is not expected to process the overload without adequate resources, but to fail in a decent manner (e.g., failure without corrupting or losing data). In stress testing the load (incoming transaction stream) is often deliberately distorted so as to force the system into resource depletion.

Whereas "Load Testing" is a test performed with an objective to determine the maximum sustainable load which the system can handle. Load is varied from a minimum (zero) to the maximum level the system can sustain without running out of resources or causing excessive delay in transactions.

<<<<< ===== >>>>>

Q. 25: What is the difference between Black Box Testing & White Box Testing?

First of all Black-Box and White-Box both are Test Design Methods.

"Black-Box" test design treats the system as a "Black-Box" (Wherein the tester can't see as to what is there inside the box). Hence we design the test cases in such a way that we pour the input from one end of the box and expect a certain specific output from the other end of the box. To run these test cases, the tester need not know as to how the input gets transformed to output inside the box. Black-Box is also known as Behavioral-Box or Functional-Box or Opaque-Box or Gray-Box or Closed-Box.

Whereas "White-Box" test design treats the system as a Transparent Box, which allows anyone to see inside the "Box". In White-Box the tester is able to see the process of transformation of an "Input" into an "Output" inside the box. Hence we design the test cases with a view to test the internal Logic, Paths or Branches of the box. White-Box is also known as Structural-Box or Glass-Box or Clear-Box or Translucent-Box test design

<<<<< ===== >>>>>

Q. 26: What is Quality?

Quality software is software that is reasonably bug-free, delivered on time and within budget, meets requirements and expectations and is maintainable. However, quality is a subjective term. Quality depends on who the customer is and their overall influence in the scheme of things.

Customers of a software development project include end-users, customer acceptance test engineers, testers, customer contract officers, customer management, the development organization's management, test engineers, testers, salespeople, software engineers, stockholders and accountants. Each type of customer will have his or her own slant on quality. The accounting department might define quality in terms of profits, while an end-user might define quality as user friendly and bug free.

<<<<< ===== >>>>>

Q. 27: What is an Inspection?

An inspection is a formal meeting, more formalized than a walkthrough and typically consists of 3-10 people including a moderator, reader (the author of whatever is being reviewed) and a recorder (to make notes in the document). The subject of the inspection is typically a document, such as a requirements document or a test plan. The purpose of an inspection is to find problems and see what is missing, not to fix anything. The result of the meeting is documented in a written report. Attendees should prepare for this type of meeting by reading through the document, before the meeting starts; most problems are found during this preparation. Preparation for inspections is difficult, but is one of the most cost-effective methods of ensuring quality, since bug prevention is more cost effective than bug detection.

<<<<< ===== >>>>>

Q. 28: What is Good Design?

Design could mean to many things, but often refers to functional design or internal design. Good functional design is indicated by software functionality can be traced back to customer and end-user requirements. Good internal design is indicated by software code whose overall structure is clear, understandable, easily modifiable and maintainable; is robust with sufficient error handling and status logging capability; and works correctly when implemented.

<<<<< ===== >>>>>

Q. 29: What is Six Sigma?

"Six Sigma" means Six Standard Deviations from the mean. It is a methodology aimed to reduce defect levels below 3.4 Defects Per one Million Opportunities. Six Sigma approach improves the process performance, decreases variation and maintains consistent quality of the process output. This leads to defect reduction and improvement in profits, product quality and customer satisfaction.

<<<<< ===== >>>>>

Q. 30: What is difference between CMM and CMMI?

"CMM" means "Capability Maturity Model" developed by the Software Engineering Institute (SEI). It is a process capability maturity model, which aids in the definition and understanding of an organization's processes. CMM is intended as a tool for objectively assessing the ability of government contractors' processes to perform a contracted software project.

Whereas "CMMI" means "Capability Maturity Model Integration" & it has superceded CMM. The old CMM has been renamed to Software Engineering CMM (SE-CMM).

<<<<< ===== >>>>>

Q. 31: What is Verification?

Verification ensures the product is designed to deliver all functionality to the customer; it typically involves reviews and meetings to evaluate documents, plans, code, requirements and specifications; this can be done with checklists, issues lists, walkthroughs and inspection meetings.

<<<<< ===== >>>>>

Q. 32: What is Validation?

Validation ensures that functionality, as defined in requirements, is the intended behavior of the product; validation typically involves actual testing and takes place after verifications are completed.

<<<<< ===== >>>>>

Q. 33: What is a Test Plan?

A software project test plan is a document that describes the objectives, scope, approach and focus of a software testing effort. The process of preparing a test plan is a useful way to think through the efforts needed to validate the acceptability of a software product. The completed document will help people outside the test group understand the why and how of product validation. It should be thorough enough to be useful, but not so thorough that none outside the test group will be able to read it.

<<<<< ===== >>>>>

Q. 34: What is a Walkthrough?

A walkthrough is an informal meeting for evaluation or informational purposes. A walkthrough is also a process at an abstract level. It's the process of inspecting software code by following paths through the code (as determined by input conditions and choices made along the way). The purpose of code walkthroughs is to ensure the code fits the purpose. Walkthroughs also offer opportunities to assess an individual's or team's competency.

<<<<< ===== >>>>>

Q. 35: What is Software Life Cycle?

Software life cycle begins when a software product is first conceived and ends when it is no longer in use. It includes phases like initial concept, requirements analysis, functional design, internal design, documentation planning, test planning, coding, document preparation, integration, testing, maintenance, updates, re-testing and phase-out.

<<<<< ===== >>>>>

Q. 36: What is the Difference between STLC & SDLC?

STLC means "Software Testing Life Cycle". It starts with activities like : 1) Preparation of Requirements Document 2) Preparation of Test Plan 3) Preparation of Test Cases 4) Execution of Test Cases 5) Analysis of Bugs 6) Reporting of Bugs 7) Tracking of Bugs till closure.

Whereas SDLC means "Software Development Life Cycle" is a software development process, used by a systems analyst to develop an information system. It starts with activities like :

- 1) Project Initiation
- 2) Requirement Gathering and Documenting
- 3) Designing

- 4) Coding and Unit Testing
- 5) Integration Testing
- 6) System Testing
- 7) Installation and Acceptance Testing
- 8) Support or Maintenance

<<<<< ===== >>>>>

Q. 37: What are the various components of STLC?

Various components of "Software Testing Life Cycle" are

- 1) Requirements Document
- 2) Preparation of Test Plan
- 3) Preparation of Test Cases
- 4) Execution of Test Cases
- 5) Analysis of Bugs
- 6) Reporting of Bugs
- 7) Tracking of Bugs till closure

<<<<< ===== >>>>>

Q. 38: What is the Difference between Project and Product Testing?

If any organization is developing the application according to the client specification then it is called as project. Accordingly its testing is known as "Project Testing"

Whereas If any organization is developing the application and marketing it is called as product. Hence its testing is known as "Product Testing"

<<<<< ===== >>>>>

Q. 39: What are the Testing Types & Techniques?

Black Box and White Box are the most popular types of software testing. These are not the stand-alone testing techniques.

Testing techniques falling under the Black-Box type are: 1) Equivalence Partitioning 2) Boundary Value Analysis 3) Cause-Effect Graphing 4) Error-Guessing etc.

Whereas testing techniques falling under the White-Box type are:

- 1) Statement coverage
- 2) Decision coverage
- 3) Condition coverage
- 4) Decision-condition coverage
- 5) Multiple condition coverage
- 6) Basis Path Testing
- 7) Loop testing
- 8) Data flow testing etc.

<<<<< ===== >>>>>

Q. 40: How do you introduce a new software QA process?

It depends on the size of the organization and the risks involved. For large organizations with high-risk projects, a serious management buy-in is required and a formalized QA process is necessary. For medium size organizations with lower risk projects, management and organizational buy-in and a slower, step-by-step process is required.

Generally speaking, QA processes should be balanced with productivity, in order to keep any bureaucracy from getting out of hand. For smaller groups or projects, an ad-hoc process is more appropriate. A lot depends on team leads and managers, feedback to developers and good communication is essential among customers, managers, developers, test engineers and testers. Regardless the size of the company, the greatest value for effort is in managing requirement processes, where the goal is requirements that are clear, complete and testable.

<<<<< ===== >>>>>

Q. 41: What are the common problems coming across Software Development Process?

Common problems in software development process are:

- 1) Poor Projection of Requirements - Generally the users are not very clear in regards to their exact needs. Most of the specifications given to Software Development Outsourcing vendors are rough and very sketchy. Problems arise if the requirements are unclear, incomplete, too general, and not testable etc.
- 2) Miscommunication - Becomes the main cause of problem when the developers remain ignorant of the exact needs or expectations of the customer.

- 3) Unrealistic Schedules - Cause problems if too much work is crammed in too little time.
- 4) Inadequate Testing - Problems arise when the application has not been adequately tested before giving it to the customer & the customer complains after using it or when there is a systems crash.

<<<<< ===== >>>>>

Q. 42: What is the role of documentation in QA?

Documentation plays a critical role in QA. QA practices should be documented, so that they are repeatable. Specifications, designs, business rules, inspection reports, configurations, code changes, test plans, test cases, bug reports, user manuals should all be documented. Ideally, there should be a system for easily finding and obtaining

of documents and determining what document will have a particular piece of information.

<<<<< ===== >>>>>

Q. 43: What is Phase Containment and Defect Prevention?

Phase Containment is incorporating QA into all the phases of SDLC. It results in Defect Prevention. If QA team performs Requirements Review, Design Review and Code Review, defects would be few when actual application is tested. That means we have prevented many defects by performing reviews at each stage of SDLC.

<<<<< ===== >>>>>

Q. 44: Why a Developer should not Test?

Of course, a developer can test, but he can't be a good tester. If developers do the testing of their own work or of the work of their peers, then the following problems crop up

- 1) Misunderstandings of the requirements or specifications may go unnoticed.
- 2) Under a given time frame, usual tendency of developers is to allocate more time in improving the code or documentation rather than doing the testing of the code.
- 3) Developers have a tendency of being optimistic of producing a defect free code hence 'under' test the product.
- 4) Testing needs great skill, while an occasional tester with no prior training in testing techniques is no match to a trained bug hunter whose sole activity is testing.
- 5) To uncover large number of bugs, tester needs to be aggressive. Whereas developer will not be aggressive, if he is testing his own product.

Testers are rewarded if they hunt lots of bugs, developers are rewarded if the product they developed has less number of bugs and this balance can only be maintained if the separate

teams exist for testing and development.

<<<<< ===== >>>>>

Q. 45: Out of Tester & Developer, who is most appropriate to do Unit Testing & Integration Testing?

Of course Developers.

It is quite difficult for a tester to do unit testing and integration testing since it involves in-depth understanding of the code. Hence developers should do the unit testing and integration testing. While doing unit testing & integration testing, a few misinterpretation of requirements might escape from the notice of the developer, but its better to test with these issues rather than not testing at all.

For a better success, code developed by one developer, then his peer should do the unit test.

<<<<< ===== >>>>>

Q. 46: What are the important Check Points for Installation Testing?

Installation testing of a new software application should be able to check points like

- 1) To check previously installed versions of the software or its dependent software or patches like previously installed Service Packs. This is to ensure that older version of the software should not get installed over the newer version.
- 2) Installer should be able to define Default Installation Path like "C:\Program Files\."
- 3) Installer should be able to allow the user to install the new software at a location different from the Default Installation Path.
- 4) To check if the product can be installed "Over the Network"
- 5) Installation should start automatically when the CD is inserted in the CD drive.
- 6) Software Remove / Repair options should be available to the user while using the Installer.
- 7) While uninstalling the software, check that all the registry keys, files, DLL, shortcuts, active X components are removed from the system.
- 8) To check if the product can be installed without Administrative Privileges (login as guest).
- 9) To check if the product can be installed on different operating systems.
- 10) To check if the product can be installed on a system having non-compliant configuration like with less Memory / RAM / Hard Disc Capacity.

<<<<< ===== >>>>>

Q. 47: What is Installation Testing?

"Installation Testing" is performed to ensure that all the Installed features and options of the software are functioning properly. Its main objective is to verify that all necessary components of the application are actually installed or not without missing out any component.

<<<<< ===== >>>>>

Q. 48: Do automated testing tools make testing easier?

For larger projects, or ongoing long-term projects, they can be valuable. But for small projects, the time needed to learn and implement them is usually not worthwhile. A common type of automated tool is the record/playback type. For example, a test engineer clicks through all combinations of menu choices, dialog box choices, buttons, etc. in a GUI and has an automated testing tool record and log the results. The recording is typically in the form of text, based on a scripting language that the testing tool can interpret. If a change is made (e.g. new buttons are added, or some underlying code in the application is changed), the application is then retested by just playing back the recorded actions and compared to the logged results in order to check effects of the change.

<<<<< ===== >>>>>

Q. 49: What should be done after a bug is found?

When a bug is found, it needs to be communicated and assigned to developers that can fix it.

After the problem is resolved, fixes should be re-tested. Additionally, determinations should be made regarding requirements, software, hardware, safety impact, etc., for regression testing to check the fixes didn't create other problems elsewhere.

If a problem-tracking system is in place, it should encapsulate these determinations. A variety of commercial, problem-tracking/management software tools are available. These tools, with the detailed input of software test engineers, will give the team complete information so developers can understand the bug, get an idea

of its severity, reproduce it and fix it.

<<<<< ===== >>>>>

Q. 50: What to do when software is full of bugs & it can't be tested at all?

In this situation the best solution is to have test engineers go through the process of reporting whatever bugs or problems initially show up, with the focus being on critical bugs.

Since this type of problem can severely affect schedules and indicates deeper problems in the software development process, such as insufficient unit testing, Insufficient integration

testing, poor design, improper build or release procedures, managers should be notified and provided with some documentation as evidence of the problem.

<<<<< ===== >>>>>

Q. 51: What is Parallel Testing?

Parallel testing involves testing multiple products or sub-components simultaneously. A parallel test station typically shares a set of test equipment across multiple test sockets, but, in some cases, it may have a separate set of hardware for each unit under test (UUT).

The majority of nonparallel test systems test only one product or sub-component at a time, leaving expensive test hardware idle more than 50 percent of the test time. Thus, with parallel testing, you can increase the throughput of manufacturing test systems without spending a lot of money to duplicate and fan out additional test systems.

<<<<< ===== >>>>>

Q. 52: What is Comparison Testing?

Comparison testing is testing that compares software weaknesses and strengths to those of competitors' products.

<<<<< ===== >>>>>

Q. 53: What is Probe Testing?

It is almost same as Exploratory testing. It is a creative, intuitive process. Everything testers do is optimized to find bugs fast, so plans often change as testers learn more about the product and its weaknesses.

Session-based test management is one method to organize and direct exploratory testing. It allows us to provide meaningful reports to management while preserving the creativity that makes exploratory testing work. This page includes an explanation of the method as well as sample session reports, and a tool we developed that produces metrics from those reports.

<<<<< ===== >>>>>

Q. 54: What questions you would ask to yourself while deciding to automate the tests?

Best approach would be to raise the following questions:

- 1) Automating this test and running it once will cost more than simply running it manually once. How much more?
- 2) An automated test has a finite lifetime, during which it must recoup that additional cost. Is this test likely to die sooner or later? What events are likely to end it?
- 3) During its lifetime, how likely is this test to find additional bugs (beyond whatever bugs it

found the first time it ran)? How does this uncertain benefit balance against the cost of automation?

<<<<< ===== >>>>>

Q. 55: What do we lose with Automation compared to Manual Testing?

Creating an automated test is usually more time-consuming & costly than running it once manually. The cost differential varies, depending on the product and the automation style.

1) If the product is being tested through a GUI and your automation style is to write scripts that drive the GUI, an automated test may be several times as expensive as a manual test.

2) If you use a GUI capture / replay tool that tracks your interactions with the product and builds a script from them, automation is relatively cheaper. It is not as cheap as manual testing, though, when you consider the cost of recapturing a test from the beginning after you make a mistake, the time spent organizing and documenting all the files that make up the test suite, the aggravation of finding and working around bugs in the tool, and so forth. Those small "in the noise" costs can add up surprisingly quickly.

3) If you're testing a compiler, automation might be only a little more expensive than manual testing, because most of the effort will go into writing test programs for the compiler to compile. Those programs have to be written whether or not they're saved for reuse.

<<<<< ===== >>>>>

Q. 56: What is the difference between Structural testing & functional testing?

Structural testing examines how the program works, taking into account possible pitfalls in the structure and logic.

Functional testing examines what the program accomplishes, without regard to how it works internally.

<<<<< ===== >>>>>

Q. 57: What is the difference between code coverage analysis & test coverage analysis?

Both these terms are similar. Code coverage analysis is sometimes called test coverage analysis. The academic world generally uses the term "test coverage" whereas the practitioners use the term "code coverage".

<<<<< ===== >>>>>

Q. 58: What are the basic assumptions behind coverage analysis?

Following assumptions tell us about the strengths and limitations of coverage analysis technique.

- 1) Bugs relate to control flow and you can expose Bugs by varying the control flow. For example, a programmer wrote "if (c)" rather than "if (!c)".
- 2) You can look for failures without knowing what failures might occur and all tests are reliable, in that successful test runs imply program correctness. The tester understands what a correct version of the program would do and can identify differences from the correct behavior.
- 3) Other assumptions are achievable specifications, no errors of omission, and no unreachable code.

<<<<< ===== >>>>>

Q. 59: What are main advantages of statement coverage metric of software testing?

- 1) The main advantage of statement coverage metric is that it can be applied directly to object code and does not require processing source code. Usually the performance profilers use this metric.
- 2) Bugs are evenly distributed through code; therefore the percentage of executable statements covered reflects the percentage of faults discovered.

<<<<< ===== >>>>>

Q. 60: What are the drawbacks of statement coverage metric of software testing?

- 1) It is insensitive to some of the control structures.
- 2) It does not report whether loops reach their termination condition - only whether the loop body was executed. With C, C++, and Java, this limitation affects loops that contain break statements.
- 3) It is completely insensitive to the logical operators (|| and &&).
- 4) It cannot distinguish consecutive switch labels.

<<<<< ===== >>>>>

Q. 61: What are advantages & drawbacks of decision coverage metric of software testing?

Decision coverage has the main advantage of simplicity & is free from many problems of statement coverage.

Disadvantage of decision coverage is that this metric ignores branches within boolean expressions which occur due to short-circuit operators.

<<<<< ===== >>>>>

Q. 62: What is multiple condition coverage metric of software testing?

Multiple condition coverage reports whether every possible combination of boolean sub-expressions occurs. 100% multiple condition coverage implies 100% condition determination coverage.

Drawback of this metric is that it becomes tedious to find out the minimum number of test cases required, especially for very complex boolean expressions. Another drawback of this metric is that the number of test cases required can vary to a large extent among various conditions having similar complexity.

<<<<< ===== >>>>>

Q. 63: What are advantages & drawbacks of path coverage metric of software testing?

Advantages are:

- 1) Path coverage requires extremely thorough testing.

Disadvantages are:

- 1) Since loops introduce an unbounded number of paths, this metric considers only a limited number of looping possibilities.
- 2) The number of paths is exponential to the number of branches. For example, a function containing 10 if-statements has 1024 paths to test. Adding just one more if-statement doubles the count to 2048.
- 3) Many paths are impossible to exercise due to relationships of data.

<<<<< ===== >>>>>

Q. 64: What is the best sequence of coverage goals as implementation strategy

- 1) Invoke at least one function in 90% of the source files (or classes).
- 2) Invoke 90% of the functions.
- 3) Attain 90% condition/decision coverage in each function.
- 4) Attain 100% condition/decision coverage.

<<<<< ===== >>>>>

Q. 65: What is configuration Management?

Configuration Management is a discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change

processing and implementation status, and verify compliance with specified requirements.

<<<<< ===== >>>>>

Q. 66: What is "Key Word Driven" or "Test Plan Driven" Method of Testing?

This method uses the actual Test Case document developed by the tester using a spreadsheet containing special "Key-Words".

In this method, the entire process is data-driven, including functionality. The Key Words control the processing.

<<<<< ===== >>>>>

Q. 67: Is the tool going to replace the testers any day?

This is not even remotely true. The automated testing tool is just another tool that will allow testers to do their jobs better by:

- 1) Performing the boring-type test cases that they now have to do over and over again
- 2) Freeing up some of their time so that they can create better, more effective test cases

The testers are still going to perform tests manually for specific application changes. Some of these tests may be automated afterward for regression testing.

<<<<< ===== >>>>>

Q. 68: What is Automated Testing?

"Automated Testing" is automating the manual testing process currently in use. This requires that a formalized "manual testing process" currently exists in your company. "Automated Testing" process includes:

- 1) Detailed test cases, including predictable "expected results", which have been developed from Business Functional Specifications and Design documentation.
- 2) A standalone Test Environment, including a Test Database that is restorable to a known constant, such that the test cases are able to be repeated each time there are modifications made to the application.

<<<<< ===== >>>>>

Q. 69: What is the purpose of Automated Test Tools?

The real use and purpose of automated test tools is to automate regression testing.

This means that we must have or must develop a database of detailed test cases that are

repeatable, and this suite of tests is run every time there is a change to the application to ensure that the change does not produce unintended consequences.

<<<<< ===== >>>>>

Q. 70: What is a Traceability Matrix?

Traceability means that we would like to be able to trace back and forth how and where any work product fulfills the directions of the preceding product.

The matrix deals with the where, while the how we have to do ourselves, once we know the where.

<<<<< ===== >>>>>

Q. 71: What is a Data Flow Diagram (DFD)?

Data Flow Diagram is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing. It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities.

<<<<< ===== >>>>>

Q. 72: What is the Difference between Retest and Regression Testing?

When a bug is fixed by the developer, testing the same bug to ensure whether it has been fixed or not is known as retesting.

Whereas testing the other features of the application which might be affected by the bug fix is known as regression testing.

<<<<< ===== >>>>>

Q. 73: What is a Test Scenario?

Test Scenario is a set of test cases which ensure that the business process flows are tested from end to end. They may be independent tests or a series of tests that follow each other, each dependent on the output of the previous one.

The terms "test scenario" and "test case" are often used synonymously.

<<<<< ===== >>>>>

Q. 74: What is Statement Coverage In software testing,?

Statement coverage is one of the ways of measuring code coverage. It describes the degree to which the software code of a program has been tested.

All the statements in the code must be executed and tested.

<<<<< ===== >>>>>

Q. 75: What is What is Defect Leakage?

Defect leakage refers to the defect Found \ reproduced by the Client or User, which the tester was unable to found.

<<<<< ===== >>>>>

Q. 76: What is the difference between Functional Testing and System Testing?

Functionality testing is based on functional requirements of the application whereas the system testing is end to end testing it covers all the functionality, performance, usability, database, stress testing.

Functional testing is the subset of system testing, but both are Black box testing.

<<<<< ===== >>>>>

Q. 77: What Is a Test Bed?

Test bed is the environment which is required to test the software.

This includes requirement of Hardware, Software, Memory, CPU speed, Operating system etc.

<<<<< ===== >>>>>

Q. 78: What is Latent Bug?

Latent Bug is a bug, which gets unobserved in two or more releases of the application.

<<<<< ===== >>>>>

Q. 79: What is Bi-directional Traceability Matrix?

Bidirectional Traceability refers to the Forward and Backward traceability.

Forward Traceability is from requirements to design to code to testcases.

Whereas the Backward Traceability is in the reverse direction, meaning thereby the end product has met the requirements or not. It is quite difficult to achieve the Backward traceability without the help of a tool.

Bidirectional Traceability is the prime expectation of CMMI.

<<<<< ===== >>>>>

Q. 80: What is Base Lining?

Baselining is the process by which the quality and cost effectiveness of a service is assessed, usually in advance of a change to the service.
Baselining usually includes comparison of the service before and after the Change or analysis of trend information.

<<<<< ===== >>>>>

Q. 81: What is configuration Management?

Configuration Management (or CM) is the processes of controlling, coordinating and tracking the Standards and procedures for managing changes in an evolving software product.

Configuration Testing is the process of checking the operation of the software being tested on various types of hardware.

<<<<< ===== >>>>>

Q. 82: What is the role of QA in a software producing company?

QA is responsible for managing, implementing, maintaining and continuously improving the Processes in the Company and enable internal projects towards process maturity and facilitate process improvements and innovations in the organization.

Tester is responsible for carrying out the testing efforts in the company.

In many companies QA person is responsible both the roles of Testing as well as creating and improving the processes.

<<<<< ===== >>>>>

Q. 83: What is Fuzz Testing?

Fuzz testing a technique of testing an application by feeding random inputs.

<<<<< ===== >>>>>

Q. 84: What is Failure Mode and Effect Analysis (FMEA)?

Failure Mode and Effect Analysis is a systematic approach to risk identification and analysis of identifying possible modes of failure and attempting to prevent their occurrence.

<<<<< ===== >>>>>

Q. 85: What is Path Testing?

Path Testing or Path Coverage is a white box method of testing which satisfies the coverage criteria through which the program is tested across each logical path. Usually, paths through

the program are grouped into a finite set of classes and one path out of every class is tested.

In Path Coverage flow of execution takes place from the start of a method to its exit. Path Coverage ensures that we test all decision outcomes independently of one another

<<<<< ===== >>>>>

Q. 86: What is Test Maturity Model or TMM?

Test Maturity Model or TMM is a five level staged framework for test process improvement, related to the Capability Maturity Model (CMM) that describes the key elements of an effective test process.

<<<<< ===== >>>>>

Q. 87: What is Back-To-Back Testing?

Back-To-Back Testing refers to the testing process in which two or more variants of a component or system are executed with the same inputs, the outputs compared, and analyzed in cases of discrepancies.

<<<<< ===== >>>>>

Q. 88: What is a Blocked Test Case?

Blocked Test Case refers to the test case, which cannot be executed because the preconditions for its execution are not fulfilled.

<<<<< ===== >>>>>

Q. 89: What is the difference between API & ABI?

Application Programming Interface (API) is a formalized set of software calls and routines that can be referenced by an application program in order to access supporting system or network services.

Whereas Application Binary Interface (ABI) is a specification defining requirements for portability of applications in binary forms across different system platforms and environments.

<<<<< ===== >>>>>

Q. 90: What is I V & V?

I V & V means Independent Verification and Validation.

Verification typically involves reviews and meetings to evaluate documents, plans, code,

requirements, and specifications. Verification can be done with the help of checklists, issues lists, walkthroughs, and inspection meetings. Whereas Validation typically involves actual testing and takes place after verifications are completed.

<<<<< ===== >>>>>

Q. 91: What is LCSAJ?

LCSAJ means "Linear Code Sequence And Jump". It consists of the following three items like:

- 1) The start of the linear sequence of executable statements.
- 2) The end of the linear sequence.
- 3) The target line to which control flow is transferred at the end of the linear sequence.

<<<<< ===== >>>>>

Q. 92: What is Measure of Completeness in software testing?

In software testing there are two measures of completeness, code coverage and path coverage.

Code coverage is a white box testing technique to determine how much of a program's source code has been tested. There are several fronts on which code coverage is measured. Code coverage provides a final layer of testing because it searches for the errors that were missed by the other test cases.

Whereas Path coverage establishes whether every potential route through a segment of code has been executed and tested.

<<<<< ===== >>>>>

Q. 93: What is Pair Programming?

Pair Programming is a software development approach whereby lines of code of a component are written by two programmers sitting at a single computer. This means ongoing real-time code reviews are performed.

<<<<< ===== >>>>>

Q. 94: What is N+1 Testing?

N+1 Testing is a variation of Regression Testing. It involves testing conducted with multiple cycles in which errors found in test cycle 'N' are resolved and the solution is re-tested in test cycle N+1. The cycles are typically repeated until the solution reaches a steady state and there are no errors.

<<<<< ===== >>>>>

Q. 95: What is Error Seeding?

Error Seeding is the process of intentionally adding known defects to those already in the component or system for the purpose of monitoring the rate of detection and removal, and estimating the number of remaining defects.

<<<<< ===== >>>>>

Q. 96: What is Extreme Programming?

Extreme Programming called XP in short is a deliberate and disciplined approach to software development.

XP is extremely successful since it lays maximum stress on the customer satisfaction. XP methodology is designed to deliver the software as per the customer needs and at a time when it is needed.

While following XP methodology the developers confidently respond to fast changing requirements of the customer may be quite late in the life cycle.

XP methodology lays emphasis on an excellent team work. Managers, customers, and developers are all part of a team dedicated to delivering quality software.

<<<<< ===== >>>>>

Q. 97: What is the difference between an application server and a Web server?

Web server serves pages for viewing in a Web browser, while an application server provides methods that client applications can call.

A little more precisely, we can say that, a Web server exclusively handles HTTP requests, whereas an application server serves business logic to application programs through any number of protocols.

<<<<< ===== >>>>>

Q. 98: What is Cause Effect Analysis?

Cause effect analysis is an approach for studying the specifications carefully and identifying the combinations of input conditions or causes and their effect in the form of a table and designing test cases.

It is suitable for applications in which combinations of input conditions are few and readily visible.

<<<<< ===== >>>>>

Q. 99: What is Error Guessing?

Error guessing is a supplementary technique of test case design involving test case design based on the tester's intuition and experience. There is no formal procedure. However, a checklist of common errors is taken for reference.

<<<<< ===== >>>>>

Q. 100: What is Basis Path Testing?

Basis Path Testing is white box testing method involving design of test cases to cover every statement, every branch and every condition in the code which has been written.

This method attempts statement coverage, decision coverage and condition coverage

<<<<< ===== >>>>>

Q. 101: What is the difference a Software Tester & Testing Analyst?

Testing analysts are more commonly involved with tasks at a higher level of abstraction, such as test process design, test planning, and test case design.

Whereas Software Testers may be involved with test case design and test procedure construction, and interaction with the actual software systems.

<<<<< ===== >>>>>

Q. 102: What are Software Testing Specialities?

Testing specialties include test automation, load testing, usability testing, testing methodology, software inspections, industry or application expertise, test metrics, test management, white box testing & security testing etc.

<<<<< ===== >>>>>

Q. 103: What can be the various Job Levels in the Software Testing Domain in a Company?

Various job levels within the testing domain can include the tester, test analyst, test manager or test specialist, test consultant or Test executive.

<<<<< ===== >>>>>

Q. 104: What is a Test Suite?

Set of collection of test cases is called a test suite.

It contains more detailed instructions or goals for each collection of test cases. It contains a section where the tester identifies the system configuration used during testing. It may also contain prerequisite states or steps, and descriptions of the tests as well.

<<<<< ===== >>>>>

Q. 105: What is a scenario test?

This is a test based on a hypothetical story used to help a person think through a complex problem or system.

Generally scenario test have following five key characteristics.

- 1) A story
- 2) Which is motivating
- 3) Which is credible
- 4) Which is complex
- 5) Which is easy to evaluate.

Scenario tests are different from test cases in a way that test cases cover single steps whereas scenarios cover a number of steps. Test suites and scenarios can be used together for a complete system test.

<<<<< ===== >>>>>

Q. 106: What is Defect Tracking?

In engineering practice, defect tracking is the process of finding defects in a product by the process of inspection, testing, or recording feedback from customers, and tracking them till their closure.

In software engineering, defect tracking is of significant importance, since complex software systems have thousands of defects due to which their management, evaluation and prioritizing is a difficult task. Hence defect tracking systems in software engineering are computer database systems which store defects and help people to manage them.

<<<<< ===== >>>>>

Q. 107: What is Formal Verification in context with Software & Hardware systems?

Formal verification is the process of proving or disproving the correctness of a system with respect to a certain formal specification or property, with the help of formal methods. Generally the formal verification is carried out algorithmically.

Approaches to implement formal verification are :

- 1) State space enumeration
- 2) Symbolic state space enumeration

- 3) Abstract interpretation
- 4) Abstraction refinement
- 5) Process-algebraic methods
- 6) Reasoning with the help of automatic theorem provers like HOL or Isabelle.

<<<<< ===== >>>>>

Q. 108: What is the concept of Fuzz Testing?

Fuzz testing is a software testing technique involving attachment of the inputs of a program to a source of random data. Main advantage of fuzz testing is that the test design is extremely simple, and remains free of preconceptions about system behavior.

Fuzz testing is generally used in large software development projects which use black box testing. Fuzz testing provides a high benefit to cost ratio.

Fuzz testing technique is also used for the measurement of quality of large software systems. The advantage is that the cost of generating tests is relatively low.

Fuzz testing helps to enhance the software security and software safety because it often finds odd oversights and defects which normal human testers would fail to find, and even the most careful human test designers would fail to create tests for.

Fuzz testing is not a substitute for exhaustive testing or formal methods; it can only provide a random sample of the system's behavior. Passing a fuzz test may only indicate that a particular software is capable to handle exceptions without crashing and it may not indicate its correct behavior.

<<<<< ===== >>>>>

Q. 109: What are the different forms of fuzz testing?

- 1) Valid fuzz Testing to assure that the random input is reasonable, or conforms to actual production data.
- 2) Simple fuzz Testing usually uses a pseudo random number generator to provide an input.
- 3) A combined approach uses valid test data with some proportion of totally random input injected.

By using all the above techniques in combination, fuzz-generated randomness can test the un-designed behavior surrounding a wider range of designed system states.

<<<<< ===== >>>>>

Q. 110: What is a Web Application & How does it look like?

A web application is an internet based application, consisting of a set many scripts, which are normally stored on some Web server and are made to interact with some databases or any other similar sources of the dynamic content.

Web applications provide an interactive Form to the user, wherein feeds inputs according to the fields provided in the form; then he clicks on a button like "Submit" or "OK" to store his inputs on the database & perform a set of calculations & present back the desired information.

Web Applications are becoming popular since these are a via media for exchange of information between various service providers and respective customers across the internet. These web applications are by & large not dependent on any platform. Popular examples of Web applications are Google / Yahoo or similar search engines, Internet Banking websites of several Banks, E-mail facility providing sites like Gmail, Yahoo Mail, Rediff Mail etc., Sale & Purchase sites like E-Bay etc.

<<<<< ===== >>>>>

Q. 111: What is Server Side Includes or SSI?

Server Side Includes or SSI is a mechanism by which we can include files using a special type of HTML comment which is similar to the include feature of todays scripting languages like JSP & PHP etc.

Old type CGI programs and ASP scripts are still using Server Side Includes or SSI to include libraries of code.

<<<<< ===== >>>>>

Q. 112: What is the difference between Dynamic Analysis & Static Analysis?

Dynamic Analysis: refers to the process of testing and evaluation of a program by executing data in real-time. The objective is to find errors in a program while it is running, rather than by repeatedly examining the code offline. Smoke testing is a type of dynamic analysis.

Static Analysis: refers to a set of techniques of program analysis where the program is not actually executed rather it is analyzed by some tools to produce the desired information. Objective of performing static analysis to ensure soundness and completeness of the program.

<<<<< ===== >>>>>

Q. 113: What is Vulnerability Analysis?

Vulnerability Analysis is a process which defines, identifies, and classifies the security holes or vulnerabilities in a computer, network, or communications infrastructure. It can be used to predict the effectiveness of proposed countermeasures and evaluate their actual effectiveness after they are implemented in actual practice.

Vulnerability Analysis or Vulnerability Assessment involves following steps:

- # Defining and classifying network or system resources
- # Assigning relative levels of importance to the resources
- # Identifying potential threats to each resource
- # Developing a strategy to deal with the most serious potential problems
- # Defining and implementing ways to minimize the consequences if an attack occurs.

<<<<< ===== >>>>>

Q. 114: What is a Vulnerability Scanner?

A vulnerability scanner is a tool for detecting & reporting genuine vulnerabilities in the system. It uses an up-to-date database containing complete information necessary to check a system for security holes. It provides facility to carry out multiple manual scans at a time. Its reports provide recommendations for countermeasures to remove the vulnerabilities detected by it.

<<<<< ===== >>>>>

Q. 115: What is an Ethical Hacker?

Ethical hacker is a computer and network expert who legitimately attacks the security system on behalf of the management with an objective to find vulnerabilities, which any malicious hacker could exploit.

For testing a security system, ethical hackers use the same methods as malicious hackers, but their aim is to report back the problems instead of taking advantage of out of them. Ethical hacking is also known as penetration testing or intrusion testing.

<<<<< ===== >>>>>

Q. 116: What is Database testing?

Database testing involves the following activities:

- 1) Testing of Data validity.
- 2) Testing of Data Integrity.
- 3) Performance testing related to the data base.
- 4) Testing of Procedure, triggers and functions.

<<<<< ===== >>>>>

Q. 117: What are the things checked in Database Testing?

Following things are generally checked In Database Testing:

- 1) Validation of field size
- 2) Checking of constraints.
- 3) Checking of Indexes as to whether done or not
- 4) Checking of stored procedures
- 5) Checking as to whether the field size defined in the application is matching with that in the database or not

<<<<< ===== >>>>>

Q. 118: What is High Order Testing?

High Order Testing is black-box testing conducted on the software after the completion of all integration activities.

<<<<< ===== >>>>>

Q. 119: What is Internationalization or I18N?

Internationalization refers to the development and testing relating to handling foreign text and data within a software program in such a way that it will be easy to adapt it to several international markets having different languages and cultures. Internationalization includes sorting, importing and exporting text and data, correct handling of currency and date and time formats, string parsing and upper / lower case handling etc.

Method of deriving the abbreviation I18N for Internationalization goes like this:

First, we take the first letter of the word Internationalization we want to abbreviate; in this case the letter "I". Next, we take the last letter in the word; in this case the letter "N". These become the first and last letters in the abbreviation. Finally, we count the remaining letters in the word between the first and last letter. In this case, "nternationalizatio" has 18 characters in it. Thus we shall encapsulate the number 18 between the "I" and "N"; thus making the final abbreviation as I18N.

<<<<< ===== >>>>>

Q. 120: What is localization or L10N?

Localization refers to development, testing and adapting the software product to suit a local or regional market. The objective of localization is to ensure suitability of the product with language & cultural aspects of the users of a particular region.

localization includes translating the program, choosing appropriate icons and graphics, and other cultural considerations. It also may include translating the program's help files and the documentation.

<<<<< ===== >>>>>

Q. 121: What is Globalization G11N?

Globalization refers to the activities performed for the purpose of marketing a software product in regional markets. The objective of globalization is to take care of global marketing accounting for economic and legal factors. The main focus of globalization is on marketing providing total enterprise solutions and a support to the management.

<<<<< ===== >>>>>

Q. 122: What are the benefits of Software Validation?

Software validation is an important tool employed to assure the quality of the software products. Few benefits are as under:

- 1) It increases the usability and reliability of the device software, resulting in reduced failure rates, less recalls and corrective actions, less liability to device manufacturers.
- 2) It reduces the long term costs by making it easier and less costly to reliably modify software and revalidate software changes.
- 3) It helps to reduce the long-term cost of software by reducing the cost of validation for each subsequent release of the software.

<<<<< ===== >>>>>

Q. 123: What is the role of Design Reviews in Software Development Life Cycle?

Design review is a primary tool for managing and evaluating software development projects. Design reviews allow management to confirm that all goals defined in the software validation plan have been achieved. Formal design reviews are more structured and include participation from others outside the development team.

Design reviews are documented, comprehensive, and systematic examinations of a design to evaluate the adequacy of the design requirements, to evaluate the capability of the design to meet these requirements, and to identify problems.

Design reviews include examination of development plans, requirements specifications, design specifications, testing plans and procedures, all other documents and activities associated with the project.

<<<<< ===== >>>>>

Q. 124: What is the need of Software Validation after a change?

When any change even a small one is made to the software, following activities need to be performed:

- 1) Re-establishment of the validation status of the software.
- 2) Conducting necessary validation analysis - not for the sake of validation of the individual change, but o to know the effect of the change on the entire software system.
- 3) Conducting suitable level of regression testing to show that unchanged but vulnerable portions of the system have not been adversely affected. Regression testing is meant to provide a confidence that the software has been validated after the change.

<<<<< ===== >>>>>

Q. 125: What is Output Forcing?

Output Forcing is a sort of functional testing of software applications.

It refers to choosing test inputs to ensure that all or the selected software outputs are generated by the testing.

<<<<< ===== >>>>>